STIC Search Report
EIC 2100

STIC Database Tracking Number: 188793

TO: Cory Bell **Location: RND 2B29** **Art Unit: 2164** **Tuesday, June 13, 2006** **Case Serial Number: 10/723369**	**From: Emory Damron** **Location: EIC 2100** **RND 4B19** **Phone: 571-272-3520** **Emory.Damron@uspto.gov**

Search Notes

Dear Cory,

Please find below an inventor search in the bibliographic and full-text foreign patent files, as well as keyword searches in the patent and non-patent literature files, both bibliographic and full text.
References of potential pertinence have been tagged, but please review all the packets in case you like something I didn't.
Of those references which have been tagged, please note any manual highlighting which I've done within the document.
In addition to searching on Dialog, I also searched EPO/JPO/Derwent, EBSCO, IEEEXplore, Google and Inspec..
There may be a few decent references contained herein, but I'll let you determine how useful they may be to you.
Please contact me if I can refocus or expand any aspect of this case, and please take a moment to provide any feedback (on the form provided) so EIC 2100 may better serve your needs. Good Luck!

Sincerely,
Emory Damron
Technical Information Specialist
EIC 2100, US Patent & Trademark Office
Phone: (571) 272-3520
Emory.damron@uspto.gov

Access DB# 188793

SEARCH REQUEST FORM

Scientific and Technical Information Center

Requester's Full Name: Cory Bell Examiner #: 81680 Date: 5/4/06
Art Unit: 2164 Phone Number 30 22736 Serial Number: 10/723369
Mail Box and Bldg/Room Location: RNO 2B-29 Results Format Preferred (circle): PAPER DISK E-MAIL

If more than one search is submitted, please prioritize searches in order of need.

****

Please provide a detailed statement of the search topic, and describe as specifically as possible the subject matter to be searched. Include the elected species or structures, keywords, synonyms, acronyms, and registry numbers, and combine with the concept or utility of the invention. Define any terms that may have a special meaning. Give examples or relevant citations, authors, etc, if known. Please attach a copy of the cover sheet, pertinent claims, and abstract.

Title of Invention: Value Mapping

Inventors (please provide full names): Hermann Burgmeier

Earliest Priority Filing Date: 12-6-2002

For Sequence Searches Only *Please include all pertinent information (parent, child, divisional, or issued patent numbers) along with the appropriate serial number.*

See Attached Note.

See Claims 1, 5, and 7

PGPUB 20040117392

SAP AKTIENGESELLSCHAFT

RECEIVED
MAY 04 2006
BY:.....

XCOPY

****

STAFF USE ONLY

STIC STIC Search Results Feedback Form

EIC 2100

Questions about the scope or the results of the search? Contact ***the EIC searcher or contact:***

Alyson Dill, EIC 2100 Team Leader
272-3527, RND 4B28

Voluntary Results Feedback Form

- *I am an examiner in Workgroup:* 2164 *Example: 2133*
- *Relevant prior art **found**, search results used as follows:*
  - ☐ 102 rejection
  - ☐ 103 rejection
  - ☐ Cited as being of interest.
  - ☐ Helped examiner better understand the invention.
  - ☐ Helped examiner better understand the state of the art in their technology.

  Types of relevant prior art found:
  - ☐ Foreign Patent(s)
  - ☐ Non-Patent Literature
    (Journal articles, conference proceedings, new product announcements etc.)
- *Relevant prior art **not found**:*
  - ☐ Results verified the lack of relevant prior art (helped determine patentability).
  - ☐ Results were not useful in determining patentability or understanding the invention.

Comments:

Drop off or send completed forms to STIC/EIC2100 RND, 4B28

sira Search and Information Resources Administration

Set Items Description
S1 2283798 S MAP? ? OR MAPP??? OR CORRELAT? OR ASSOCIAT? OR AFFILIAT? OR MATCH? OR LINK? OR LOCATE? OR LOCATING OR ADDRESS?
S2 1717343 S DIRECTING OR TUNNEL? OR POINTING? OR CORRESPOND? OR EQUATE? OR EQUATING? OR ROUTE? OR ROUTING
S3 859391 S SOFTWARE()OBJECT? OR DATABASE? OR DOMAIN? OR COLUMN? OR TABLE? OR DATAOBJECT? OR DATASTRUCTUR? OR DATAVALUE?
S4 755106 S OBJECT? OR DATAOBJECT? OR DATA()(ENTIT? OR STRUCTUR?) OR OBJECTMODEL? OR OBJECT()MODEL?
S5 6603 S DATACOMMAND? OR DATAORDER? OR DATAUNIT? OR DATA()UNIT? OR DATA()MESSAG? OR DATA?()INSTRUCT
S6 85215 S DATABYTE? OR DATABLOCK? OR DATA()(BLOCK? OR REQUEST? OR INPUT? OR TYPE? OR MEMBER? OR VALUE?) OR DATATYPE?
S7 66730 S (DATA? OR SOFTWARE OR OBJECT?)(2N)(IDENTIFIER? OR TAG? ? OR METADATA? OR ADDRESS? OR LOCATER? OR LABEL? OR NAME? OR EXTENSION? OR DESIGNATION?)
S8 210523 S DATA()(REPRESENTATION? OR DEPOSITOR? OR REPOSITOR? OR BANK? OR FILE? OR COLLECTION?) OR ROW OR ROWS
S9 1358351 S LOG OR LOGS OR CATALOG? OR CHRONICL? OR RECORD? ? OR LIST??? OR FIELD? OR ARRAY? OR ARCHIV?
S10 778746 S S1:S2 AND S3:S9
S11 113719 S GUI? ? OR (GRAPHIC? OR VISUAL?)(2W)(INTERFACE? OR APP? ? OR APPLICAT?) OR DISPLAY? OR MENU? ? OR WEBPAGE?
S12 211367 S COMPUTER?()(SCREEN? OR WINDOW?) OR SHOW? ? OR SHOWED OR SHOWING OR DROPDOWN? OR COMPUTER?(2N)(IMPLEMENT? OR TOOL? ? OR UTILIT?)
S13 561 S USERINTERFAC? OR (DROP OR PULL)()DOWN? OR (WINDOW OR DESKTOP)()MANAGER? OR PULLDOWN?
S14 7772 S UI OR UIS OR (GRAPHIC? OR VISUAL? OR INFORMATION? OR DATA?)(2N)(MENU? OR PAGE? OR SCREEN?)
S15 173771 S PLURAL? OR MULTIP? OR MULTIT? OR SEVERAL? OR MORE(2W)ONE OR MANY OR NUMEROUS?
S16 167493 S COLLECTION? OR NETWORK? OR ARRAY? OR ASSEMBLY? OR ASSEMBLIE?
S17 143234 S MODULAT? OR CONVERT? OR MODIF? OR ALTER? OR TRANSFORM?
S18 113677 S CHANGE? OR CHANGING? OR TRANSLAT? OR AMEND? OR EDIT? OR UPDAT?
S19 37042 S CONVERSION? OR MANIPULAT? OR CUSTOMIZ? OR PERSONALIZ? OR INDIVIDUALIZ?
S20 52133 S CUSTOMIS? OR ADAPT? OR RECONFIGUR? OR CONFIGUR? OR REVIS? OR REARRANG?
S21 232 S PERSONALIS? OR (CUSTOM OR TAILOR)()(MAKE? OR MADE? OR MAKING)
S22 7854 S REORDER? OR RESHUFFL? OR RESET? OR REFORMAT? OR READDRESS?
S23 353799 S WITHOUT OR LESS OR "NOT" OR MISSING? OR NO OR UN OR NON OR LACKING OR ABSENT OR FREE OR MINUS OR ELIMINAT? OR SKIP? OR OMIT?
S24 336 S COULDN? OR WOULDN? OR SHOULDN? OR DOESN? OR CAN()T OR WON()T
S25 972 S UNMODULAT? OR NONMODULAT? OR UNCONVER? OR NONCONVER? OR UNMODIF? OR NONMODIF? OR NONALTER? OR UNALTER? OR NONTRANSFORM? OR UNTRANSFORM?
S26 1405 S UNCHANG? OR NONCHANG? OR IMMUTAB? OR NONTRANSLAT? OR UNTRANSLAT? OR UNAMEND? OR NONAMEND? OR UNEDIT? OR NONEDIT? OR NONUPDAT? OR UNUPDAT?
S27 8 S UNMANIPULAT? OR NONMANIPULAT? OR UNCUSTOM? OR NONCUSTOM? OR UNPERSONALI? OR NONPERSONALI? OR NONINDIVIDUALI? OR UNINDIVIDUALI?
S28 28 S UNADAPT? OR NONADAPT? OR UNRECONFIGUR? OR NONRECONFIG? OR NONCONFIGUR? OR UNCONFIGUR? OR UNREVIS? OR NONREVIS? OR NONREARRANG? OR UNREARRANG?
S29 3 S MODULATIONLESS? OR MODULATIONFREE? OR CHANGEFREE? OR CHANGELESS? OR MODIFICATIONLESS? OR MODIFICATIONFREE?
S30 0 S AMENDMENTLESS? OR AMENDMENTFREE? OR EDITFREE? OR EDITLESS? OR UPDATELESS? OR UPDATEFREE? OR REVISIONLESS? OR REVISIONFREE?
S31 67728 S SOURCE? OR ISSUER? OR ISSUING? OR SEMINAL?
S32 112951 S NUMBER()(ONE OR 1) OR PRINCIPAL? OR LEAD OR CONTROLLER? OR HEAD OR MASTER OR BASIC?
S33 192868 S FIRST? OR 1ST OR PRIMARY OR INITIAL? OR ORIGINAL? OR LEADOFF? OR MAIN OR CHIEF OR

```
INTRODUCTORY?
S34   193924  S TARGET? OR RECEIV? OR RECEPT? OR RECIPIENT? OR DESTINATION? OR ENDPOINT?
S35   168300  S SECOND? OR 2ND OR ANOTHER OR AUXILIAR? OR BACKUP? OR EXTRA OR SLAVE? OR
SUPPLEMENT?
S36   104672  S SUBSIDIAR? OR DIFFERENT? OR ALTERNAT? OR NUMBER()(TWO OR 2)
S37   201119  S IC=G06F?
S38   160049  S MC=T01?
S39        0  S S10 AND S3:S9(5N)17:S22 AND S1:S2(5N)(S25:S30 OR S23:S24(3N)S17:S22)
S40    27272  S S10 AND S17:S22 AND (S25:S30 OR S23:S24(5N)S17:S22)
S41    11231  S S40 AND S11:S14
S42     2787  S S40 AND S1:S2(10N)S3:S9(10N)(S31:S33 AND S34:S36)
S43     1061  S S41 AND S42
S44     3753  S S40:S43 AND S1:S2(10N)(S25:S30 OR S23:S24(3N)S17:S22)
S45     1976  S S44 AND S3:S9(10N)S17:S22
S46      507  S S44 AND S42
S47      127  S S41 AND S42 AND S44 AND S45
S48      127  S S46 AND S47
S49      191  S S43 AND S46
S50      191  S S48:S49
S51      128  S S50 AND S37:S38
S52      191  S S50:S51
S53      100  S S52 AND AC=US/PR
S54       85  S S53 AND AY=(1970:2002)/PR
S55       74  S S53 NOT AY=(2003:2006)/PR
S56       91  S S52 NOT S53
S57       64  S S56 AND PY=1970:2002
S58       52  S S56 NOT PY=2003:2006
S59      149  S S54:S55 OR S57:S58
S60      149  IDPAT (sorted in duplicate/non-duplicate order)
S61    14782  S S41:S45
S62     1120  S S61 AND (MAP? ? OR MAPP???)
S63      700  S S62 AND S37:S38
S64      300  S S63 AND (S43:S46)
S65      279  S S64 NOT S52
S66      107  S S65 AND (S23:S24(3N)S17:S22 OR S25:S30)(5N)(MAP? ? OR MAPP???)
S67      115  S S64 AND (S23:S24(3N)S17:S22 OR S25:S30)(5N)(MAP? ? OR MAPP???)
S68      115  S S66:S67
S69       38  S S68 AND AC=US/PR
S70       29  S S69 AND AY=(1970:2002)/PR
S71       28  S S69 NOT AY=(2003:2006)/PR
S72       77  S S68 NOT S69
S73       38  S S72 AND PY=1970:2002
S74       30  S S72 NOT PY=2003:2006
S75       67  S S70:S71 OR S73:S74
S76       67  IDPAT (sorted in duplicate/non-duplicate order)
S77      210  S S60 OR S76
; show files
```

[File 347] **JAPIO** Dec 1976-2005/Dec(Updated 060404)

[File 350] **Derwent WPIX** 1963-2006/UD,UM &UP=200634

**File 350: Preview the enhanced DWPI through ONTAP DWPI (File 280). For more information, visit http://www.dialog.com/dwpi/.*

77/3,K/51 (Item 26 from file: 350) **Links**
Derwent WPIX

015930539 **Image available**
WPI Acc No: 2004-088380/200409
XRPX Acc No: N04-070750

Persistent target identification provision apparatus for fiber channel environment, sorts lists of physical address identifiers by permanent unique identifiers comprising world wide node name

Patent Assignee: LSI LOGIC CORP (LSIL-N)
Inventor: ODENWALD L
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 6671727	B1	20031230	US 99467461	A	19991220	200409	B

Priority Applications (No Type Date): US 99467461 A 19991220
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6671727	B1		9	G06F-013/00	

Persistent target identification provision apparatus for fiber channel environment, sorts lists of physical address identifiers by permanent unique identifiers comprising world wide node name

Abstract (Basic):
... A memory circuit stores a **list** of physical **address** identifiers of target devices (106a-106n), which are sorted by permanent unique identifiers comprising world wide node name such that the position of each physical **address** identifier remains constant. The communication between host (102) and the target devices, is **not** affected by **change** in the physical **address** identifier.
... Allows **mapping** of fiber channel node names into logical name **table** and allows the **mapping** to remain **unchanged**, even if the physical **address changes**.
...

...The figure **shows** the block diagram of the communication path
...Title Terms: **LIST**;
International Patent Class (Main): **G06F-013/00**
Manual Codes (EPI/S-X): **T01-N02A3B**...

US006671727B1

(12) **United States Patent**
Odenwald

(10) **Patent No.: US 6,671,727 B1**
(45) **Date of Patent: *Dec. 30, 2003**

(54) **METHODOLOGY FOR PROVIDING PERSISTENT TARGET IDENTIFICATION IN A FIBRE CHANNEL ENVIRONMENT**

(75) Inventor: **Louis Odenwald**, Witchita, KS (US)

(73) Assignee: **LSI Logic Corporation**, Milpitas, CA (US)

(*) Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/467,461**

(22) Filed: **Dec. 20, 1999**

(51) **Int. Cl.**[7] **G06F 13/00**

(52) **U.S. Cl.** **709/227**; 709/228; 709/229; 709/245; 370/351

(58) **Field of Search** 709/227, 228, 709/229, 245; 370/351

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,598,541	A		1/1997	Malladi
5,768,530	A		6/1998	Sandorfi
5,842,040	A		11/1998	Hughes et al.
5,848,251	A		12/1998	Lomelino et al.
5,954,796	A	*	9/1999	McCarty et al. 709/222
5,956,723	A		9/1999	Zhu 707/100
5,978,379	A	*	11/1999	Chan et al. 370/403
5,983,309	A	*	11/1999	Atsatt et al. 711/4
6,057,863	A		5/2000	Olarig
6,130,892	A	*	10/2000	Short et al. 370/401
6,138,161	A	*	10/2000	Reynolds et al. 709/227
6,141,731	A		10/2000	Beardsley
6,195,703	B1	*	2/2001	Blumenau et al. 709/238
6,195,705	B1	*	2/2001	Leung 709/245
6,199,112	B1	*	3/2001	Wilson 709/227
6,269,099	B1	*	7/2001	Borella et al. 370/389
6,401,128	B1	*	6/2002	Stai et al. 709/236
6,418,476	B1	*	7/2002	Luciani 709/238
6,421,711	B1	*	7/2002	Blumenau et al. 709/213
6,421,723	B1	*	7/2002	Tawil 709/224
6,421,753	B1	*	7/2002	Hoese et al. 710/305
6,425,034	B1		7/2002	Steinmetz
6,430,645	B1	*	8/2002	Basham 710/305
6,484,200	B1	*	11/2002	Angal et al. 709/224
6,510,450	B1	*	1/2003	Ricart et al. 709/203
6,578,087	B1	*	6/2003	Garakani et al. 709/242

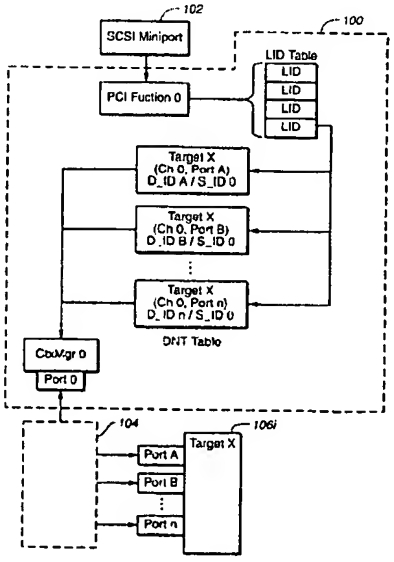
* cited by examiner

Primary Examiner—Robert B. Harrell
Assistant Examiner—M H P
(74) *Attorney, Agent, or Firm*—Christopher P. Maiorana, PC

(57) **ABSTRACT**

An apparatus comprising a first circuit and a second circuit. The first circuit may be configured to (i) generate a sorted list of permanent unique identifiers according to predetermined criteria and (ii) associate a logical identification with a physical address identifier using the sorted list. The second circuit may be configured to manage communications between a host and a target. The second circuit may (i) communicate with the host using the logical identification and (ii) communicate with the target using the physical address identifier. The communications between the host and the target may be unaffected by changes in the physical address identifier. The function of the first circuit and/or the second circuit may be implemented, among other examples, in software and/or firmware.

22 Claims, 4 Drawing Sheets



example where the circuit **108** may add the LID for a particular target device **106***a*–**106***n* to a service parameter block of the particular target device. FIG. **4**(*b*) illustrates an example where the circuit **108** may maintain a list of LIDs and a list of service parameter blocks **120***a*–**120***n*. The LIDs may be associated to the respective WWN using address pointers to the respective service parameter block. FIG. **4**(*c*) illustrates an example where the circuit **108** may maintain separate lists for the LIDs, the WWNs, and the service parameter blocks **120***a*–**120***n* that may contain the physical address identifiers. Address pointers may be used to form the associations between the LIDs, the WWNs, and the physical address identifiers. However, other methods of forming the associations may be implemented to meet the design criteria of a particular application.

Referring to FIG. **5**, a flow chart of an example start up (e.g., loop initialization) operation of the circuit **100** is shown. Upon power up or loop initialization, the circuit **100** generally runs through a process to discover all target devices **106***a*–**106***n* connected to the loop/fabric **104** (e.g., block **200**). The target devices **106***a*–**106***n* are generally sorted by WWN (e.g., block **202**) and assigned LIDs relevant to the sorted order of the WWNs (e.g., block **204**). The LID to WWN mappings generally remain constant between power cycles of the circuit **100**, regardless of the WWN to physical address identifier relationships. When a new target device **106***a*–**106***n* is added to the loop (e.g., yes path of block **206**), the WWN of the new target device is generally not sorted into the list. The WWN of the new target device is generally appended to the end of the sorted list of WWNs (e.g., block **208**). An unused LID is generally associated with the WWN and the physical address identifier of the new target device (e.g., block **210**).

In the example mentioned above, where two independent loops are merged together, all LID to WWN mappings in the circuit **100** of the initial loop will generally remain unchanged, and unused LIDs will generally be assigned to the targets on the loop being merged in, regardless of any physical address identifier changes that might take place.

In addition, LID to WWN mappings will generally remain constant across Initiator power cycles as long as no target devices **106***a*–**106***n* are added to or removed from the loop **104**. The WWNs of the target devices **106***a*–**106***n* connected to the loop **104** are generally sorted to handle the case where (i) no target devices were added to or removed from the loop **104** and (ii) physical address identifiers have changed due to target devices having been physically moved or other initiators added to or removed from the loop **104**. The LID to WWN mappings will generally remain unchanged. Since the WWN is generally used to associate the LIDs and the physical address identifiers of the target devices **106***a*–**106***n*, the LID to target mapping will generally also remain unchanged regardless of a physical address change.

Referring to FIG. **6**, a flow chart illustrating an alternative method that may provide a fixed LID for one or more devices, including a boot device, is shown. An alternative embodiment of the present invention may allow for one or more devices, that may include a boot device, to persistently regain the same LID. The circuit **100** may comprise a small amount of non-volatile memory. The LID and WWN of one or more devices, including the boot device, may be stored in the non-volatile memory. The LID and WWN of a boot device will generally be stored in the non-volatile memory before any other device. After the circuit **100** has (i) powered up (e.g., block **300**) and (ii) discovered and sorted the WWNs of the target devices **106***a*–**106***n* (e.g., blocks **302** and **304**), the WWNs and the desired LIDs may be read from the non-volatile memory (e.g., block **306**). The WWNs from the non-volatile memory may be inserted into the sorted list at a position determined by the corresponding LID from the non-volatile memory (e.g., block **308**). The process of adding new target devices **106***a*–**106***n* is generally similar to the process described in connection with FIG. **3** (e.g., blocks **312**–**316**). The host **102** is generally able to boot using a particular target device on the loop **104**, regardless of new target devices **106***a*–**106***n* being added to or removed from the loop **104**. The amount of non-volatile memory may be adjusted to balance the number of fixed LIDs and cost.

Although the present invention finds particular application in a Fibre Channel environment, it is not limited thereto. The method according to the present invention is applicable to any computing system, environment or standard where each target device of the system is assigned a unique permanent identifier, and must exchange that permanent identifier with other target devices of the system in order to exchange information. The present invention is applicable to any environment in which information, whether called login service parameters, WWNs or some other names, regarding particular devices and modules must be exchanged before communication can take place.

The function performed by the circuit **100** of FIG. **1** and or described in the flow diagrams of FIGS. **5** and **6** may be implemented using a conventional general purpose digital computer programmed according to the teachings of the present specification, as will be apparent to those skilled in the relevant art(s). Appropriate software coding can readily be prepared by skilled programmers based on the teachings of the present disclosure, as will also be apparent to those skilled in the relevant art(s).

The present invention may also be implemented by the preparation of ASICs, FPGAs, or by interconnecting an appropriate network of conventional component circuits.

The present invention thus may also include a computer product which may be a storage medium including instructions which can be used to program a computer to perform a process in accordance with the present invention. The storage medium can include, but is not limited to, any type of disk including floppy disk, optical disk, CD-ROM, and magneto-optical disks, ROMs, RAMs, EPROMS, EEPROMS, Flash memory, magnetic or optical cards, or any type of media suitable for storing electronic instructions.

While the invention has been particularly shown and described with reference to the preferred embodiments thereof, it will be understood by those skilled in the art that various changes in form and details may be made without departing from the spirit and scope of the invention. For example, the present invention may be implemented along with one or more portions of U.S. Pat. No. 5,956,723, which is hereby incorporated by reference in its entirety.

What is claimed is:

1. An apparatus comprising:

a first circuit configured to (i) store a sorted list of physical address identifiers of one or more target devices in a volatile storage medium, said list being sorted by permanent unique identifiers of said one or more target devices such that a position of each of said physical address identifiers in said list remains constant across power cyclings of said apparatus as long as said one or more target devices are unchanged and (ii) associate a logical identification with each physical address identifier of said sorted list; and

a second circuit configured to (i) couple a host to said one or more target devices, (ii) obtain said permanent unique identifiers and physical address identifiers of

said one or more target devices and generate said sorted list, (iii) communicate with said host using said logical identification and (iv) communicate with said one or more target devices using said physical address identifiers, wherein said communications between said host and said one or more target devices are unaffected by changes in said physical address identifier.

2. The apparatus according to claim 1, wherein said apparatus comprises a Fibre Channel initiator.

3. The apparatus according to claim 1, wherein said first circuit further comprises non-volatile memory.

4. The apparatus according to claim 3, wherein said non-volatile memory contains said logical identification and said permanent unique identifier of one or more of said target devices.

5. The apparatus according to claim 1, wherein said permanent unique identifier comprises a World Wide Node Name.

6. The apparatus according to claim 1, wherein said position in said list remains unchanged when said physical address identifier changes.

7. The apparatus according to claim 1, wherein said one or more target devices comprise an FC-FCP device.

8. The apparatus according to claim 1, wherein said physical address identifiers comprise one or more of an arbitrated loop physical address (AL_PA) and a destination address (D_ID).

9. The apparatus according to claim 1, wherein said first circuit comprises a volatile memory, wherein said logical identification, said permanent unique identifiers, and said physical address identifiers are stored and associated by memory pointers.

10. The apparatus according to claim 1, wherein said apparatus is connected to said one or more target devices by a Fibre Channel Arbitrated Loop (FC-AL).

11. The apparatus according to claim 1, wherein said apparatus is connected to said one or more target devices by a Fibre Channel Fabric to Loop Attachment (FC-FLA).

12. An apparatus comprising:

means for (i) storing a sorted list of physical address identifiers of one or more target devices in a volatile storage medium, said list being sorted by permanent unique identifiers of said one or more target devices such that a position of each of said physical address identifiers in said list remains constant across power cyclings of said apparatus as long as said one or more target devices are unchanged and (ii) associating a logical identification with each physical address identifier of said sorted list; and

means for (i) coupling a host to said one or more target devices, (ii) obtaining said permanent unique identifiers and physical address identifiers of said one or more target devices and generating said sorted list, (iii) communicating with said host using said logical identification and (iv) communicating with said one or more target devices using said physical address identifiers, wherein said communications between said host and said one or more target devices are unaffected by changes in said physical address identifiers.

13. A method for providing persistent logical identification and communication in a network environment comprising the steps of:

(A) storing a sorted list of physical address identifiers of one or more target devices in a volatile storage medium, wherein said list is sorted by permanent unique identifiers of said one or more target devices such that a position of each of said physical address identifiers in said list remains constant across power cyclings of said apparatus as long an said one or more target devices are unchanged and a logical identification is associated with each physical address identifier of said sorted list; and

(B) (i) coupling a host to said one or more target devices, (ii) obtaining said permanent unique identifiers and physical address identifiers of said one or more target devices and generating said sorted list, (iii) communicating with said host using said logical identification and (iv) communicating with said one or more target devices using said physical address identifiers, wherein said communications between said host and said one or more target devices are unaffected by changes in said physical address identifiers.

14. The method according to claim 13, wherein step A comprises the sub-steps of:

(A-1) retrieving said permanent unique identifiers and corresponding physical address identifiers from one or more targets in said environment;

(A-2) generating said sorted list of said permanent unique identifiers according to said predetermined criteria;

(A-3) associating said logical identification to each of said permanent unique identifiers;

(A-4) associating said logical identification to said corresponding physical address identifiers of said permanent unique identifiers; and

(A-5) when one or more new targets are added to said environment, appending a permanent unique identifier of each of said one or more new targets to said sorted list, associating an unused logical identification to each appended permanent unique identifier and corresponding physical address identifier of said one or more new targets.

15. The method according to claim 13, wherein each of said permanent unique identifiers comprises a World Wide Node Name.

16. The method according to claim 13, wherein each of said physical address identifiers changes.

17. The method according to claim 13, wherein said network comprises a Fibre Channel Arbitrated Loop (FC-AL).

18. The method according to claim 13, wherein said network comprises a Fibre Channel Fabric to Loop Attachment (FC-FLA) standard network.

19. The method according to claim 13, wherein the association of said logical identification to one of said physical address identifiers is based on a position of a corresponding one of said permanent unique identifier in said sorted list.

20. The method according to claim 14, further comprising the sub-steps of:

(A-2a) reading a permanent unique identifier and a logical identification of one or more target devices from a non-volatile memory; and

(A-2b) inserting said permanent unique identifier of said one or more target devices into a position in said sorted list such that said permanent unique identifier of said one or more devices is associated with the same logical identification as was retrieved from said non-volatile memory.

21. The apparatus according to claim 1, wherein said association between said logical identification and each of said physical address identifiers is persistent between power cyclings of said apparatus.

22. The apparatus according to claim 1, wherein said association between said logical identification and each of said physical address identifiers is persistent across a power cycling of said apparatus when no target devices have been added or removed since a previous power cycling.

* * * * *

77/3,K/171 (Item 146 from file: 350)
Derwent WPIX

011089681 **Image available**
WPI Acc No: 1997-067606/199707
XRPX Acc No: N97-055603
Multiprogramming computer system - changes ownership data associated with information and passes physical memory addresses across domain instead of virtual address

Patent Assignee: SUN MICROSYSTEMS INC (SUNM)
Inventor: CHU H K J; CHU H J
Number of Countries: 006 Number of Patents: 004
Patent Family:
Patent No Kind Date Applicat No Kind Date Week
EP 752661 A1 19970108 EP 96304665 A 19960625 199707 B
JP 9050404 A 19970218 JP 96173672 A 19960703 199717
US 5895499 A 19990420 US 95497816 A 19950703 199923
US 6256657 B1 20010703 US 95497816 A 19950703 200140
US 99231319 A 19990113

Priority Applications (No Type Date): US 95497816 A 19950703; US 99231319 A
19990113
Patent Details:
Patent No Kind Lan Pg Main IPC Filing Notes
EP 752661 A1 E 18 G06F-012/10
Designated States (Regional): DE FR GB IT
JP 9050404 A 15 G06F-012/10
US 5895499 A G06F-015/163
US 6256657 B1 G06F-017/00 Cont of application US 95497816
Cont of patent US 5895499

... **changes ownership data associated with information and passes physical memory addresses across domain instead of virtual address**

...Abstract (Basic): The computer system includes a network **adapter** which performs checksum operations. A computer and an operating system support multiple processes and multiple **domains**. A portion of a main memory is arranged in pages. A virtual memory management unit controls reads and writes to the memory using **page mapping information**. An operating system controls ownership of the pages in a memory using **page** ownership **information**. A physical **page** of **data** is

transferred between **domains** by reassigning ownership information without updating the page mapping information until the **domain** obtaining ownership needs to read the data...

...with existing network interfaces and hardware. Maintains data protection
and security. Does not require extensive **page** remapping. Minimises **data** transfers...
...Title Terms: **CHANGE**;
International Patent Class (Main): **G06F-012/10**...

...**G06F-015/163**...

...**G06F-017/00**
International Patent Class (Additional): **G06F-009/00**...

...**G06F-009/46**...

...**G06F-013/00**
Manual Codes (EPI/S-X): **T01-F05G**...

...**T01-H03A**...

...**T01-H08**

US006256657B1

(12) **United States Patent**
Chu

(10) **Patent No.: US 6,256,657 B1**
(45) **Date of Patent: *Jul. 3, 2001**

(54) **CROSS-DOMAIN DATA TRANSFER USING DEFERRED PAGE REMAPPING**

(75) Inventor: **Hsiao-keng J. Chu**, Palo Alto, CA (US)

(73) Assignee: **Sun Microsystems, Inc.**, Palo Alto, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/231,319**

(22) Filed: **Jan. 13, 1999**

Related U.S. Application Data

(63) Continuation of application No. 08/497,816, filed on Jul. 3, 1995, now Pat. No. 5,895,499.

(51) **Int. Cl.**[7] **G06F 17/00**; G06F 9/00; G06F 9/46

(52) **U.S. Cl.** **709/1**; 709/301

(58) **Field of Search** 709/1, 310–323

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,758,951	7/1988	Sznyter, III	364/200
5,088,031	2/1992	Takasaki et al.	395/400
5,179,675	1/1993	Cole et al.	395/400
5,390,310	2/1995	Welland	395/400
5,430,842	7/1995	Thompson et al.	395/200
5,522,045	5/1996	Sandberg	395/200.08

OTHER PUBLICATIONS

Dalton et al, "Afterburner", IEEE digital library, pp. 36–43, Jul. 1993.*

Peter Steenkiste, "A systematic approach to host interface design for high-speed networks", IEEE digital library, pp. 47–57, Mar. 1994.*

Druschel et al, "Experiences with a high-speed network adaptor: a software perspective", ACM digital library, pp. 2–13, 1994.*

Barrera, Joseph, "A Fast Mach Network IPC Implementation", Proc USENIX 2nd Mach Symposium, pp. 1–12, Nov. 1991.

Boykin et al., "Programming Under Mach", Chap. 4, pp. 99–148, Sep. 1993.

Druschel, Peter and Peterson, Larry, "Fbufs: A High-Bandwidth Cross-Domain Transfer Facility", *Proceedings of the Fourteenth ACM Symposium on Operating Systems Principles*, Dec. 1993.

Robinson et al., "Domain-based access control for distributed computing systems", Software Engineering Journal, pp. 161–170, Sep. 1988.

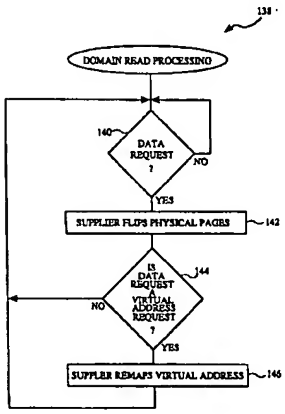
(List continued on next page.)

Primary Examiner—Majid Banankhah
Assistant Examiner—P. G. Caldwell
(74) *Attorney, Agent, or Firm*—Beyer Weaver & Thomas, LLP

(57) **ABSTRACT**

A cross-domain data transfer technique is disclosed in which page remapping operations are eliminated in situations where physical memory addresses can be passed across domains. By passing physical memory addresses across domains instead of virtual memory addresses, the page remapping operations necessarily associated with passing virtual memory addresses across domains can be avoided in many cases. With the receipt of data across domains, page remapping operations are able to be deferred until the data is received in a domain that needs to touch the data. In certain cases, the transfer of data can be completed without ever having to map in the data to the receiving domain's address space. With the transmission of data across domains, where possible the pages are borrowed in their physical form. The invention can be embodied in many ways, including system, apparatus or method forms.

18 Claims, 7 Drawing Sheets



the domain-1 **148**, domain-2 **152** and domain-3 **156** designated by reference numeral **178** illustrates the end result after the unlocking operation; namely, PAGE_C **174-4** is now again exclusively controlled by domain-3 **156**.

Thus, it is possible with the invention to receive data from a network I/O subsystem and transfer it to a disk I/O subsystem without ever having to touch the data. With the receipt of data across domains, the remapping operations are able to be deferred until the data is received in a domain that needs to touch the data. In certain cases, the transfer of data can be completed without ever having to map in the data to the receiving domain's address space. With the transmission of data across domains, where possible the pages are borrowed in their physical form.

The many features and advantages of the present invention are apparent from the written description, and thus, it is intended by the appended claims to cover all such features and advantages of the invention. Further, since numerous modifications and changes will readily occur to those skilled in the art, it is not desired to limit the invention to the exact construction and operation as illustrated and described. Hence, all suitable modifications and equivalents may be resorted to as falling within the scope of the invention.

What is claimed is:

1. A computer system, comprising:

a network adapter that connects to a network, said network adapter including circuitry for performing data checksum operations;

a computer and an operating system arranged to support multiple processes and multiple domains;

a main memory having at least a portion thereof arranged in pages; and

a virtual memory management unit, operatively connected to said computer and said memory, for managing reads and writes to said memory using page mapping information,

wherein said operating system controls ownership of the pages in said memory using page ownership information, and

wherein said computer system enables the receipt of a physical page of data transferred between domains by reassigning ownership of the physical page within the page ownership information without updating the page mapping information associated with said virtual memory management unit until the domain obtaining ownership needs to read the data.

2. A computer system as recited in claim **1**, wherein the page mapping information includes data for translating virtual memory addresses to physical addresses.

3. A computer system as recited in claim **1**, wherein said virtual memory management unit comprises a translation mapping device for mapping virtual memory addresses to physical addresses.

4. A computer system as recited in claim **1**, wherein said operating system has physical buffers and virtual buffers associated with each of the multiple domains, the physical buffers point to said main memory and the virtual buffers point to virtual memory.

5. A computer system as recited in claim **4**, wherein said network adapter receives incoming packets of data, each packet includes a header and data, and

wherein the header is placed in one of the virtual buffers and the data is placed one or more of the physical buffers.

6. A computer system as recited in claim **1**, wherein said computer system enables the transmission of a physical page of data across domains by borrowing the physical page.

7. A computer system as recited in claim **6**, wherein said computer system further enables the transmission of the physical page across domains by locking the page ownership information associated with the borrowed page while the physical page is being borrowed.

8. A computer system as recited in claim **7**, wherein the original owner of the borrowed page is restricted to read-only access while the physical page is being borrowed.

9. A computer system as recited in claim **6**, wherein the transfer of the page of data is invoked by a data request,

wherein said operating system comprises means for determining whether the requested page of data being received is one of a virtual address request and a physical page memory request,

wherein when the requested page is the physical page memory request, said operating system controls the reassigning of ownership of the physical page within the page ownership information without updating the page mapping information, and

wherein when the requested page is the virtual address request, said operating system controls the reassigning of ownership of the physical page within the page ownership information and said operating system together with said virtual memory management unit update the page mapping information.

10. A computer system as recited in claim **6**, wherein said operating system has physical buffers and virtual buffers associated with each of the multiple domains, the physical buffers point to said main memory and the virtual buffers point to virtual memory.

11. A computer system as recited in claim **10**, wherein said network adapter receives incoming packets of data, each packet includes a header and data, and

wherein the header is placed in one of the virtual buffers and the data is placed one or more of the physical buffers.

12. A computer readable medium including computer program code for transferring a block of data associated with a first domain to a second domain, the first and second domains are associated with a computer system that supports multiple domains and virtual memory, said computer readable medium comprising:

computer program code for receiving a request at a first domain from a second domain for a block of data residing on a portion of physical memory assigned to the first domain;

computer program code for reassigning ownership of the block of data requested from the first domain to the second domain;

computer program code for determining whether the block of data requested is one of a virtual address request and a physical memory request; and

computer program code for remapping virtual addresses in accordance with said computer program code for reassigning when said computer program code for determining determines that the data request is the virtual address request and not when said computer program code for determining determines that the data request is the physical memory request,

wherein burdensome remapping operations are avoided or at least deferred until said computer program code for determining determines that a data request is a virtual address request for the reassigned block of data.

13. A computer readable medium as recited in claim **12**, wherein the physical memory includes a plurality of pages, and the portion of the physical memory storing the block of data is one or more pages of the physical memory.

14. A computer readable medium as recited in claim 12, wherein the block of data is one or more pages of the physical memory, and the physical memory request is a physical page memory request.

15. A computer readable medium as recited in claim 14, wherein said computer program code for reassigning of ownership comprises computer program code for flipping the physical page requested at the first domain with a physical page of the second domain.

16. A computer readable medium as recited in claim 12, wherein the virtual address request is a read request that will need to touch the data, and the physical memory request is a read request that will not need to touch the data.

17. A computer readable medium as recited in claim 16, wherein the remapping performed by said computer program code for remapping is deferred until a read request for the data being transferred needs to touch the data.

18. A computer readable medium as recited in claim 16, wherein the first domain is a network domain and the second domain is a kernal domain.

* * * * *

US005895499A

United States Patent [19]

Chu

[11] Patent Number: 5,895,499

[45] Date of Patent: Apr. 20, 1999

[54] **CROSS-DOMAIN DATA TRANSFER USING DEFERRED PAGE REMAPPING**

[75] Inventor: **Hsiao-keng J. Chu**, Palo Alto, Calif.

[73] Assignee: **Sun Microsystems, Inc.**, Palo Alto, Calif.

[21] Appl. No.: **08/497,816**

[22] Filed: **Jul. 3, 1995**

[51] **Int. Cl.**[6] **G06F 15/163**

[52] **U.S. Cl.** **711/202**; 711/203; 395/680

[58] **Field of Search** 395/400, 200.01–200.08, 395/680, 681–685; 711/202, 203; 364/200

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,758,951	7/1988	Sznyter	364/200
5,088,031	2/1992	Takasaki et al.	395/400
5,179,675	1/1993	Cole et al.	395/400
5,390,310	2/1995	Welland	395/400
5,430,842	7/1995	Thompson et al.	395/200
5,522,045	5/1996	Sandberg	395/200.08

OTHER PUBLICATIONS

Barrera, Joseph, "A Fast Mach Network IPC Implementation", Proc USENIX 2nd Mach Symposium, pp. 1–12, Nov. 91.

Robinson et al, "Domain–based access control for distributed computing systems", *Software Engineering* pp. 161–170 Sep. '88.

Boykin et al, *Programming under Mach*, Chap 4 pp. 99–148, Sep. 93.

S.–Y. Tzou and D.P. Anderson, "The Performance of Message–passing using Restricted Virtual Memory Remapping", *Software–Practice and Experience*, vol. 21(3), Mar. 1991, pp. 251–267.

Peter Druschel and Larry L. Peterson, "Fbufs: A High–Bandwidth Cross–Domain Transfer Facility", *Proceedings of the Fourteenth ACM Symposium on Operating Systems Principles*, Dec. 1993.

James P.G. Sterbenz and Gurudatta M. Parulkar, "AXON: A High Speed Communication Architecture for Distributed Applications," *Proceedings of IEEE INFOCOM '90*, Jun., 1990.

Primary Examiner—Majid A. Banankhah

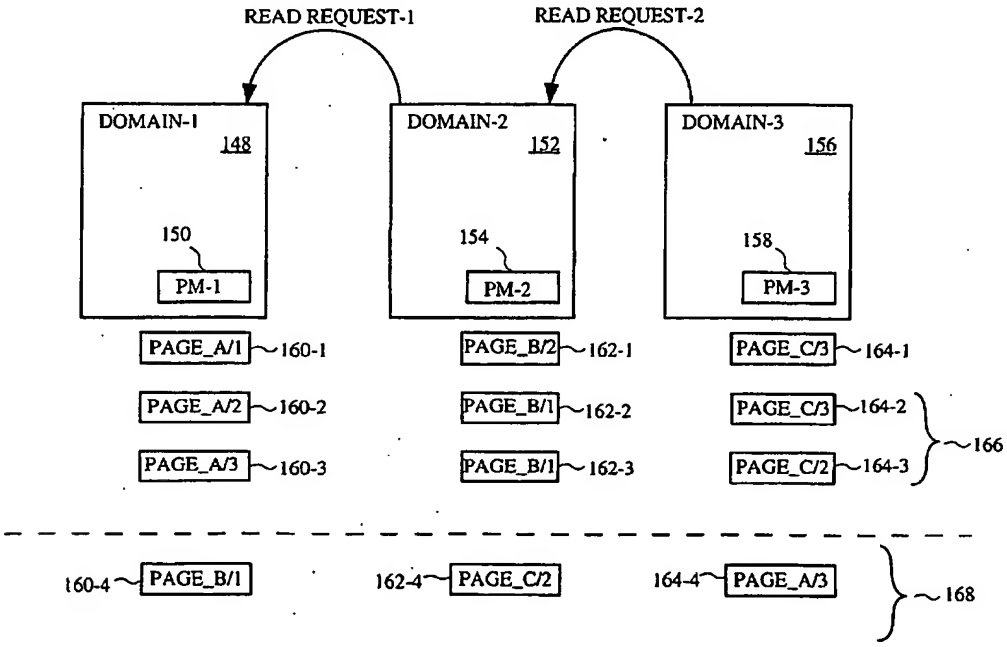
Assistant Examiner—Patricia Caldwell

Attorney, Agent, or Firm—Beyer & Weaver, LLP

[57] **ABSTRACT**

A cross-domain data transfer technique is disclosed in which page remapping operations are eliminated in situations where physical memory addresses can be passed across domains. By passing physical memory addresses across domains instead of virtual memory addresses, the page remapping operations necessarily associated with passing virtual memory addresses across domains can be avoided in many cases. With the receipt of data across domains, page remapping operations are able to be deferred until the data is received in a domain that needs to touch the data. In certain cases, the transfer of data can be completed without ever having to map in the data to the receiving domain's address space. With the transmission of data across domains, where possible the pages are borrowed in their physical form. The invention can be embodied in many ways, including system, apparatus or method forms.

17 Claims, 7 Drawing Sheets



after the unlocking operation; namely, PAGE_C 174-4 is now again exclusively controlled by domain-3 156.

Thus, it is possible with the invention to receive data from a network I/O subsystem and transfer it to a disk I/O subsystem without ever having to touch the data. With the receipt of data across domains, the remapping operations are able to be deferred until the data is received in a domain that needs to touch the data. In certain cases, the transfer of data can be completed without ever having to map in the data to the receiving domain's address space. With the transmission of data across domains, where possible the pages are borrowed in their physical form.

The many features and advantages of the present invention are apparent from the written description, and thus, it is intended by the appended claims to cover all such features and advantages of the invention. Further, since numerous modifications and changes will readily occur to those skilled in the art, it is not desired to limit the invention to the exact construction and operation as illustrated and described. Hence, all suitable modifications and equivalents may be

What is claimed is:

1. A computer system, comprising:

a network adapter that connects to a network, said network adapter including circuitry for performing data checksum operations;

a computer and an operating system arranged to support multiple processes and multiple domains;

a main memory having at least a portion thereof arranged in pages; and

a virtual memory management unit, operatively connected to said computer and said memory, for managing reads and writes to said memory using page mapping information,

wherein said operating system controls ownership of the pages in said memory using page ownership information,

wherein said computer system enables the receipt of a physical page of data transferred between domains by reassigning ownership of the physical page within the page ownership information without updating the page mapping information associated with said virtual memory management unit until the domain obtaining ownership needs to read the data, such that burdensome page remapping operations are be avoided or at least deferred until needed,

wherein the transfer of the page of data is invoked by a data request,

wherein said operating system comprises means for determining whether the requested page of data being received is one of a virtual address request and a physical page memory request,

wherein when the requested page is the physical page memory request, said operating system controls the reassigning of ownership of the physical page within the page ownership information without updating the page mapping information, and

wherein when the requested page is the virtual address request, said operating system controls the reassigning of ownership of the physical page within page ownership information and said operating system together with said virtual memory management unit update the page mapping information.

2. A computer system as recited in claim 1, wherein the page mapping information includes data for translating virtual memory addresses to physical addresses.

3. A computer system as recited in claim 1, wherein said virtual memory management unit comprises a translation mapping device for mapping virtual memory addresses to physical addresses.

4. A computer system as recited in claim 1, wherein said operating system has physical buffers and virtual buffers associated with each of the multiple domains, the physical buffers point to said main memory and the virtual buffers point to virtual memory.

5. A computer system as recited in claim 4, wherein said network adapter receives incoming packets of data, each packet includes a header and data, and

wherein the header is placed in one of the virtual buffers and the data is placed one or more of the physical buffers.

6. A computer system as recited in claim 1, wherein said computer system enables the transmission of a physical page of data across domains by borrowing the physical page.

7. A computer system as recited in claim 6, wherein said computer system further enables the transmission of the physical page across domains by locking the page ownership information associated with the borrowed page while the physical page is being borrowed.

8. A computer system as recited in claim 7, wherein the original owner of the borrowed page is restricted to read-only access while the physical page is being borrowed.

9. A computer system as recited in claim 6, wherein said operating system has physical buffers and virtual buffers associated with each of the multiple domains, the physical buffers point to said main memory and the virtual buffers point to virtual memory.

10. A computer system as recited in claim 9, wherein said network adapter receives incoming packets of data, each packet includes a header and data, and

wherein the header is placed in one of the virtual buffers and the data is placed one or more of the physical buffers.

11. In a multitasking computer system supporting multiple domains and virtual memory, a method for transferring a block of data associated with a first domain to a second domain, said method comprising the steps of:

(a) receiving a request at a first domain from a second domain for a block of data residing on a portion of physical memory assigned to the first domain;

(b) reassigning ownership of the block of data requested from the first domain to the second domain;

(c) determining whether the block of data requested is one of a virtual address request and a physical memory request; and

(d) remapping virtual addresses in accordance with said reassigning step (b) when said determining step (c) determines that the data request is the virtual address request and not when said determining step (c) determines that the data request is the physical memory requests,

wherein burdensome remapping operations are avoided or at least deferred until said determining step (c) determines that a data request is a virtual address request for the reassigned block of data.

12. A method as recited in claim 11, wherein the physical memory includes a plurality of pages, and the portion of the physical memory storing the block of data is one or more pages of the physical memory.

13. A method as recited in claim 11, wherein the block of data is one or more pages of the physical memory, and the physical memory request is a physical page memory request.

14. A method as recited in claim 13, wherein said reassigning step (b) of ownership comprises flipping the physical page requested at the first domain with a physical page of the second domain.

15. A method as recited in claim 11, wherein the virtual address request is a read request that will need to touch the data, and the physical memory request is a read request that will not need to touch the data.

16. A method as recited in claim 15, wherein said remapping step (d) is deferred until a read request for the data being transferred needs to touch the data.

17. A method as recited in claim 15, wherein the first domain is a network domain and the second domain is a kernal domain.

* * * * *

77/3,K/108 (Item 83 from file: 350)
Derwent WPIX

013972649 **Image available**
WPI Acc No: 2001-456862/**200149**
XRPX Acc No: N01-338585
Translation type mapping in gateway signaling network, involves referring mapping table for mapping translation type of local signaling network into that of non-local signaling network
Patent Assignee: LG ELECTRONICS INC (GLDS); KINSEISHA KK (GLDS); LG INFORMATION & COMMUNICATIONS LTD (GLDS)
Inventor: LEE S U; LEE S W
Number of Countries: 004 Number of Patents: 006
Patent Family:
Patent No Kind Date Applicat No Kind Date Week
US 20010008532 A1 20010719 US 2000736366 A 20001215 200149 B
JP 2001211258 A 20010803 JP 2000380164 A 20001214 200150
CN 1300167 A 20010620 CN 2000135737 A 20001215 200159
KR 2001056727 A 20010704 KR 9958315 A 19991216 200206
KR 331468 B 20020409 KR 9958315 A 19991216 200267
US 6993019 B2 20060131 US 2000736366 A 20001215 200610

Priority Applications (No Type Date): KR 9958315 A 19991216
Patent Details:
Patent No Kind Lan Pg Main IPC Filing Notes
US 20010008532 A1 10 H04L-012/28
JP 2001211258 A 7 H04M-007/00
CN 1300167 A H04Q-003/00
KR 2001056727 A H04Q-003/42
KR 331468 B H04Q-003/42 Previous Publ. patent KR 2001056727
US 6993019 B2 H04L-012/50

Translation type mapping in gateway signaling network, involves referring mapping table for mapping translation type of local signaling network into that of non-local signaling network

Abstract (Basic):
... The **translation** type information of **non**-local signaling network is defined in a **mapping table**. **Translation** type of that network, in the SCCP message received by adjacent non-signaling network is **mapped** by referring the **table**. The **translation** type of local signaling network contained in SCCP message sent to adjacent non-local signaling network is **mapped** into **translation** type of **non**-local signaling network by referring the **table**.
... For **mapping** different **translation** type in gateway

signaling network used to provide global title **translation** service, signaling connection control part (SCCP) service...

...Provides interworking SCCP service **without** a **modification** of a signaling network and suspension of service as desired by efficient **mapping.**
...

...The figure **shows** the flowchart illustrating **translation** type **mapping** definition process
Title Terms: **TRANSLATION;**
Manual Codes (EPI/S-X): **T01-H07C...**

...**T01-S01B**

US006993019B2

(12) **United States Patent**
Lee

(10) **Patent No.: US 6,993,019 B2**
(45) **Date of Patent: Jan. 31, 2006**

(54) **METHOD FOR MAPPING TRANSLATION TYPE IN NO. 7 GATEWAY SIGNALING NETWORK**

(75) Inventor: **Seong Woo Lee**, Seoul (KR)

(73) Assignee: **LG Electronics Inc.**, Seoul (KR)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 852 days.

(21) Appl. No.: **09/736,366**

(22) Filed: **Dec. 15, 2000**

(65) **Prior Publication Data**

US 2001/0008532 A1 Jul. 19, 2001

(30) **Foreign Application Priority Data**

Dec. 16, 1999 (KR) 1999-58315

(51) **Int. Cl.**
H04L 12/50 (2006.01)

(52) **U.S. Cl.** **370/385**; 370/467; 370/522; 370/401; 379/230

(58) **Field of Classification Search** 370/401, 370/524, 466–467, 522, 385, 384; 379/229–230
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,282,244	A *	1/1994	Fuller et al.	379/230
5,852,660	A *	12/1998	Lindquist et al.	379/230
5,940,492	A *	8/1999	Galloway et al.	379/230
6,094,578	A *	7/2000	Purcell et al.	455/426.1

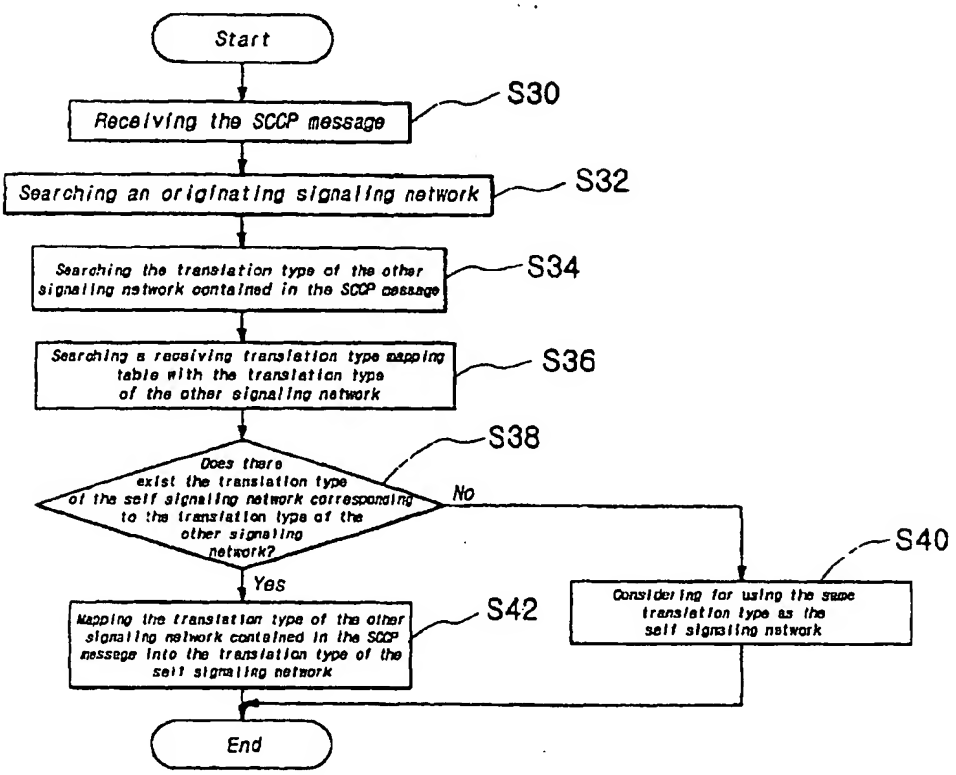
* cited by examiner

Primary Examiner—Ricky Ngo
Assistant Examiner—Nittaya Juntima
(74) *Attorney, Agent, or Firm*—Fleshner & Kim, LLP

(57) **ABSTRACT**

A method for performing the translation type mapping in the No. 7 gateway signaling network includes defining translation type information on a non-local signaling network in a translation type mapping table according to a manager's request; mapping a translation type of the non-local signaling network contained in the SCCP message received by the neighboring (adjacent) non-local signaling network by searching the translation type mapping table; and mapping a translation type of a local signaling network contained in the SCCP message transmitted to the adjacent non-local signaling network into the translation type of the non-local signaling network by searching the translation type mapping table.

15 Claims, 4 Drawing Sheets



modification of a signaling network, a suspension of a service and a novel definition of a translation type.

The foregoing embodiments and advantages are merely exemplary and are not to be construed as limiting the present invention. The present teaching can be readily applied to other types of apparatuses. The description of the present invention is intended to be illustrative, and not to limit the scope of the claims. Many alternatives, modifications, and variations will be apparent to those skilled in the art. In the claims, means-plus-function clauses are intended to cover the structures described herein as performing the recited function and not only structural equivalents but also equivalent structures.

What is claimed is:

1. A method for mapping a translation type in a No. 7 gateway signaling network, comprising:

defining translation type information of a first signaling network in a translation type mapping table;

mapping a translation type contained in a signaling connection control part (SCCP) message of the first signaling network received from an adjacent signaling network into a translation type of a second network by searching the translation type mapping table;

mapping a translation type of the second signaling network contained in a SCCP message to be transmitted to the adjacent signaling network into the translation type of the first signaling network by searching the translation type mapping table; and

inserting the mapped translation type into a same field of a protocol used to communicate between the first and second signaling networks such that a structure of the protocol is not changed.

2. The method of claim 1, wherein the translation type mapping table comprises a receiving translation type table configured to resolve the translation type of the second signaling network with at least one of a translation type of an originating signaling network transmitting the SCCP message and the translation type of the first signaling network contained in the SCCP message.

3. The method of claim 1, wherein the translation type mapping table comprises a transmitting translation type table configured to resolve a translation type of a terminating signaling network with at least one of a translation type of the terminating signaling network to receive the SCCP message and the translation type of the second signaling network contained in the SCCP message.

4. The method of claim 1, wherein defining the translation type information comprises:

receiving translation type mapping information according to a request to define a translation type mapping for the first signaling network;

storing the translation type mapping information in the translation type mapping table; and

transferring the translation type mapping information to a processor to perform a translation type mapping function.

5. The method of claim 4, wherein if the second signaling network is defined as a gateway signaling network, the translation type used by the first signaling network is defined in a SCCP signaling network, and the first signaling network is defined in the signaling network of the gateway.

6. The method of claim 4, wherein the translation type mapping information comprises the second signaling network translation type information.

7. The method of claim 4, wherein the translation type mapping information comprises information related to the first signaling network as a mapping object.

8. The method of claim 4, wherein the translation type mapping information comprises the translation type information of the first signaling network as a mapping object.

9. The method of claim 1, wherein mapping the translation type of the message received from the first network comprises:

searching an originating signaling network transmitting the SCCP message if the SCCP message is received from a signal link interworked with the adjacent signaling network, and searching the translation type contained in the SCCP message of the first signaling network if the originating signaling network is the first signaling network;

determining whether the translation type of the second signaling network corresponding to the translation type of the first signaling network exists by searching the receiving translation type mapping table with the resolved translation type of the first signaling network; and

mapping the translation type of the first signaling network contained in the SCCP message into the translation type of the second signaling network, if the translation type of the second signaling network corresponding to the translation type of the first signaling network exists.

10. The method of claim 1, wherein mapping the translation type of the message to be transmitted comprises:

searching the translation type of the second signaling network contained in the SCCP message to be transmitted if a terminating signaling network is the first signaling network;

determining whether the translation type of the first signaling network corresponding to the translation type of the second signaling network exists by searching the transmitting translation type mapping table with a resolved translation type of the second signaling network; and

mapping the translation type of the second signaling network contained in the SCCP message to be transmitted into the translation type used by the terminating network, if the translation type of the first signaling network corresponding to the translation type of the second signaling network exists.

11. A method of transmitting a signaling connection control part (SCCP) message from a first network to a second network, comprising:

generating a first SCCP signal having a first translation type;

transmitting the first SCCP signal from a first network;

searching a translation type mapping table for a definition corresponding to the first translation type;

receiving the first SCCP signal by a second network having a second translation type;

mapping the first translation type to the second translation type according to the definition from the translation type mapping table; and

inserting the mapped second translation type into a same octet as the first translational type of a protocol used to communicate between the first and second networks.

12. A method of mapping a translation type in a common channel signaling network, comprising:

identifying a first translation type of a first network based on a signaling connection control part (SCCP) message;

searching a look-up table for a second translation type of a second network corresponding to the first translation type;

mapping the first translation type to the second translation type in accordance with a definition of the look-up table; and

wherein the first and second networks use the same protocol structure to communicate with each other and the mapped second translation type is placed in a same position of the protocol as the first translation type.

13. The method of claim 12, wherein the look-up table comprises one of a receiving translation type mapping table and a transmitting translation type mapping table.

14. The method of claim 13, wherein the transmitting translation type mapping table is configured to resolve a translation type of a terminating signaling network with at least one of a translation type of the terminating signaling network to receive the SCCP message and the translation type of the second network contained in the SCCP message.

15. The method of claim 13, wherein the receiving translation type mapping table is configured to resolve the translation type of the second network with at least one of a translation type of an originating signaling network transmitting the SCCP message and the translation type of the first signaling network contained in the SCCP message.

* * * * *

77/3,K/159 (Item 134 from file: 350)
Derwent WPIX

012299261 **Image available**
WPI Acc No: 1999-105367/199909
XRPX Acc No: N99-076100
Address mapping gateway for use in internetwork link - has pair of translation units which substitutes domain network address and gateway-mapped node address in place of network number and node address and vice versa, respectively
Patent Assignee: NOVELL INC (NOVE-N)
Inventor: ALLEN M D; GERVAIS J L; NATARAJAN A E; PERLMAN R J
Number of Countries: 001 Number of Patents: 001
Patent Family:
Patent No Kind Date Applicat No Kind Date Week
US 5856974 A 19990105 US 96600773 A 19960213 199909 B

Priority Applications (No Type Date): US 96600773 A 19960213
Patent Details:
Patent No Kind Lan Pg Main IPC Filing Notes
US 5856974 A 16 H04L-012/66

Address mapping gateway for use in internetwork link - ...

...has pair of translation units which substitutes domain network address and gateway-mapped node address in place of network number and node address and vice versa, respectively

...Abstract (Basic): The gateway **receives** a **first** data packet with a network layer header including a network layer **address**, from a **source** node (342) within a **domain**. The network layer **address** includes a node **address** and network number of a local network which includes the **source** node. A gateway-mapped node address that is unique within the domain, is generated. An **address mapping table** with several **mapping** entries, each of which is **associated** with a node in the **domain**. Each **mapping** entry includes the node **address**, the network number and the **corresponding** gateway-**mapping** bode ad- dress. A first translation unit substitutes the **domain** network **address** and the gateway-**mapped** node **address** in place of the network number and the node **address** in a packet generated by a node in the **domain**, respectively. A **second** data packet including network layer header with network layer **address**, is received in the **destination** node in the **domain**. The network layer

address in the **second** packet includes gateway-**mapped** node **address** and the **domain** network **address**. A **second** translation unit substitutes the network number and the node **address** in the **second** packet, in place of the **domain** network **address** and the gateway-**mapped** node **address**, respectively...

...ADVANTAGE - Maintains **translation** exemption **table** containing **non-translatable** network numbers by gateway, so as to prevent **translation** of certain local network **addresses** that is to be made available to backbone. Enables **address mapping** gateway to explicitly advertise **non-translatable** network numbers, rather than **translating addresses** of nodes on **non-translatable** network number. **Eliminates** need for backbone to maintain **different** network number for each local network in the internetwork. Ena- bles **addressing** to be compatible with any intra-**domain** and inter-**domain routing** protocol...
Title Terms: **ADDRESS**;
Manual Codes (EPI/S-X): **T01-H01A**...

US005856974A

United States Patent [19]

Gervais et al.

[11] Patent Number: 5,856,974

[45] Date of Patent: Jan. 5, 1999

[54] INTERNETWORK ADDRESS MAPPING GATEWAY

[75] Inventors: Joseph L. Gervais, Fremont; Alampoondi E. Natarajan; Michael D. Allen, both of San Jose, all of Calif.; Radia J. Perlman, Acton, Mass.

[73] Assignee: Novell, Inc., Orem, Utah

[21] Appl. No.: 600,773

[22] Filed: Feb. 13, 1996

[51] Int. Cl.[6] H04L 12/66

[52] U.S. Cl. 370/392; 370/401; 370/475

[58] Field of Search 370/389, 390, 370/392, 393, 400, 401, 466, 402, 467, 404, 471, 405, 475, 409; 395/200.01, 200.02, 200.03, 200.16, 200.2, 410, 412, 415, 416, 421.11, 200.48, 200.53, 200.6, 200.68, 200.75, 200.79

[56] References Cited

U.S. PATENT DOCUMENTS

4,933,937	6/1990	Konishi	370/404
4,933,938	6/1990	Sheehy	370/401
5,150,464	9/1992	Sidhu et al.	395/200.1
5,159,592	10/1992	Perkins	370/388
5,163,045	11/1992	Caram et al.	370/409
5,249,181	9/1993	Wang et al.	370/409
5,313,465	5/1994	Perlman et al.	370/254
5,390,173	2/1995	Spinney et al.	370/401
5,428,615	6/1995	Backes et al.	370/392
5,442,633	8/1995	Perkins et al.	370/331
5,636,216	6/1997	Fox et al.	370/402

OTHER PUBLICATIONS

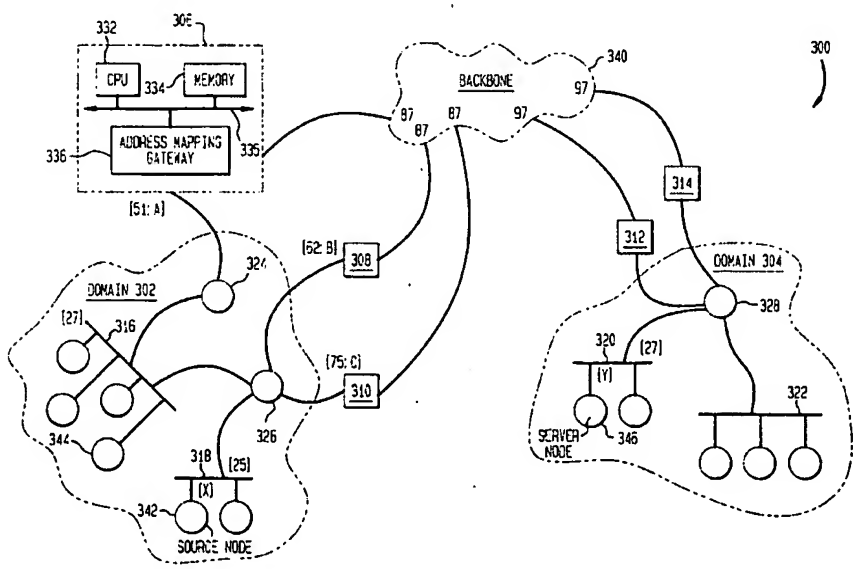
Apple Computer, "Appletalk Update-Based Routing Protocol: Enhanced Appletalk Routing", Aug. 1993, pp. 1-50.

Primary Examiner—Chi H. Pham
Assistant Examiner—Kwang B. Yao
Attorney, Agent, or Firm—Cesari and McKenna, LLP

[57] ABSTRACT

The present invention is an address mapping gateway, used in an internetwork link, that associates all nodes in a domain with a single network number (referred to as a domain network address), and provides gateway-mapped node addresses that are unique within the domain. The address mapping gateway dynamically substitutes the "globally-unique" domain network address and the "domain-unique" gateway-mapped node address for a network number and node address, respectively, of a network layer address of a packet header received from a source node in the domain. Conversely, when a packet is received for a destination node in the domain, the address mapping gateway substitutes the originally-assigned network number and node address for the domain network address and gateway-mapped node address, respectively, prior to forwarding the packet to the node. Specifically, the address mapping gateway maintains an address mapping table that provides a cross-reference between (1) a source node address and the network number of the local network in which the node resides, and (2) a gateway-mapped node address generated by the address mapping gateway. Upon receipt of a packet from the source node, the address mapping gateway creates a mapping entry in the address mapping table that node and performs an address translation to a globally-unique network layer address. On the other hand, upon receipt of a packet destined for a destination node in the domain, the address mapping gateway locates the mapping entry for the packet's gateway-mapped node address and performs a reverse address translation to the originally-assigned network layer address.

14 Claims, 6 Drawing Sheets



Address mapping gateway **308** then forwards the packet to gateways **306** and **310**, replacing the domain network address in the network address number field **402** with the internal network address 51:A and the gateway-mapped node address in one message and 75:C in another. Address mapping gateway **308** will then discard the packet. Address mapping gateway **306** will discard the packet since it does not have a mapping entry for a node address of Z. However, gateway **310** does have such an entry and, upon receipt, will replace its internal network address of 75:C in the network address field **402** with the network number **25** of the local network **318** and the node address X of node **342**, and forward the packet to destination node **342**. Gateway **310** also sends a mapping entry update message to gateway **308**, providing it with the association of network address 25:X to the gateway-mapped node address Z. The gateway **308** then updates its address mapping table **500** accordingly.

It should be understood that embodiments of the present invention can be implemented in hardware, software or a combination thereof. In such functional components, the steps they perform would be implemented in hardware and/or software to perform the functions of the present invention. Furthermore, the present invention may be implemented in a standardized software program stored in any type of computer-readable medium. Any presently available or future developed computer software language can be employed in such embodiments of the present invention. Furthermore, the present invention may be implemented in dedicated circuits of conventional design with those skilled in the art. Any presently available or future developed circuitry can be employed in such embodiments of the present invention.

Furthermore, the terms and expressions which have been employed are used as terms of description and not of limitation, and there is no intention, in the use of such terms and expressions, of excluding any equivalents of the features shown and described or portions thereof, but it is recognized that various modifications are possible within the scope of the invention claimed.

What is claimed is:

1. An address mapping gateway for use in an internetwork link, for representing substantially all nodes in a domain as being associated with a domain network address, comprising:

means for receiving a first packet from a source node within the domain, said first packet having a network layer header including a network layer address, said network layer address comprising a node address and a network number of a local network including said source node;

means for generating a gateway-mapped node address that is unique within the domain;

an address mapping table having one or more mapping entries, each said mapping entry associated with a node within the domain and including said node address, said network number, and said gateway-mapped node address;

means for creating said mapping entries in said address mapping table;

first translation means for substituting the domain network address in place of said network number and for substituting said gateway-mapped node address in place of said node address in a packet generated by a node in the domain;

means for receiving a second packet for a destination node in the domain, said second packet having a network layer header including a network layer address, said network layer address comprising said gateway-mapped node address and said domain network address; and

second translation means for substituting said network number and said node address in place of said domain network address and said gateway-mapped node address, respectively, of said second packet.

2. In an internetwork having two or more local networks interconnected by one or more internetwork links, a method for addressing nodes in an internetwork link, said method comprising the steps of:

(a) generating a domain network address to represent substantially all the nodes in a domain, wherein said domain network address is a unique network number within the internetwork;

(b) receiving a packet generated by a source node within said domain, said packet including a source node address header field having a source network number of a local network including said source node, and a source node address of said source node;

(c) generating a gateway-mapped node address for said source node, wherein said gateway-mapped node address is unique within said domain, said step of generating comprising the steps of

(1) determining whether said source node address is unique within said domain;

(2) assigning a first value to said gateway-mapped node address equivalent to said source node address when said source node address is unique within said domain;

(3) assigning a second value to said gateway-mapped node address when said source node address is not unique within said domain, wherein said second value is a node address value unique within said domain; and

(d) substituting said domain network address and said gateway-mapped node address in place of said source network number and said source node address, respectively, in said source node address header field.

3. In an internetwork link of an internetwork, a method for representing substantially all nodes in a domain as being associated with a single network number, the method comprising the steps of:

(a) receiving a packet from a first node within the domain, said packet having a network layer header with a first network layer address, said network layer address including a network number of a local network in which said first node resides, and an originally-assigned node address of said first node;

(b) translating said first network layer address to a second network layer address, wherein said second network layer address is unique within the internetwork, and said step of translating includes the steps of

(1) generating a domain network number of said second network layer address, wherein said domain network number is unique within the internetwork;

(2) generating a gateway-mapped node address of said second network layer address, wherein said gateway-mapped node address is unique within the domain, and comprising the steps of

a) determining whether said node address is unique within the internetwork;

b) setting said gateway-mapped node address to said node address when said node address is a unique node address;

c) combining said network number of said local network and said originally-assigned node address to create a domain-unique gateway-mapped node address value when said node address is not unique within the domain; and

(3) replacing said first network layer address with said second network layer address in said network layer header of said packet.

4. An address mapping gateway for associating substantially all nodes in a domain with a domain network address, comprising:

means for receiving a packet from a first node within the domain, said packet having a network layer header including a network layer address, said network layer address comprising a node address of said first node and a network number of a local network including said first node;

means for providing a gateway-mapped node address for said first node, wherein said gateway-mapped node address is unique within the domain, and said means for providing includes

a) means for determining whether said node address is unique within the domain;

b) means for assigning a first value to said gateway-mapped node address equal to said node address when said node address is unique within the domain, and for assigning a second value to said gateway-mapped node address that is unique within the domain when said node address is not unique within the domain;

means for substituting the domain network address for said network number and for substituting said gateway-mapped node address for said node address in said packet;

means for receiving a second packet, destined for said first node, said second packet having a second network layer header with a second network layer address, said second network layer address including said domain network address identifying said local network including said first node and said gateway-mapped node address to identify said first node; and

means for substituting said network number and said node address in place of said domain network address and said gateway-mapped node address of said second packet, respectively.

5. The address mapping gateway of claim 4, wherein the domain is accessible through a plurality of internetwork links each having an address mapping gateway, wherein each said address mapping gateway comprises:

means for forwarding a packet to other of said plurality of address mapping gateways when said address mapping gateway does not have a mapping entry associated with a destination node identified by a network layer address of said packet; and

means for providing one or more other address mapping gateways with a mapping entry from said address mapping table.

6. In an internetwork having two or more domains interconnected by one or more internetwork links, a method for addressing nodes in an internetwork link to a domain which comprises a plurality of local networks, comprising the steps of:

(a) generating a domain network address to represent substantially all the nodes in the domain, wherein said domain network address is a unique network number within the internetwork;

(b) receiving a packet generated by a source node within said domain, said packet including a source node address header field having a source network number of a local network including said source node, and a source node address of said source node;

(c) referencing a translation exemption table containing non-translatable network numbers and performing steps (d) and (e) when said translation exemption table does not contain said source network number;

(d) generating a gateway-mapped node address for said source node, wherein said gateway-mapped node address is unique within said domain;

(e) substituting said domain network address and said gateway-mapped node address in place of said source network number and said source node address, respectively, in said source node address header field;

(f) creating a mapping entry into an address mapping table having one or more mapping entries, said mapping entry associated with said source node and including a node address field for storing said source node address, a network number field for storing said source network number, and a mapped node address field for storing said gateway-mapped node address.

7. The method of claim 6, further comprising the steps of:

(a) receiving a packet for a destination node in said domain, said packet having a destination node address header field including said domain network address and said gateway-mapped node address; and

(b) substituting a destination network number of a second local network within said domain and a destination node address of a node within said second local network in place of said domain network address and said gateway-mapped node address, respectively.

8. The method of claim 6 further comprising the step of advertising said non-translatable network numbers into the internetwork.

9. The method of claim 6 further comprising the steps of:

(a) receiving, at a first internetwork link of said one or more internetwork links, a packet for a destination node in said domain, said packet having a destination node address header having a network number equivalent to said domain network address, and a destination node address not equivalent to a gateway-mapped node address in a first address mapping table of said first internetwork link;

(b) forwarding said packet to a second internetwork link of said plurality of internetwork links;

(c) searching gateway-mapped node addresses in a second address mapping table in said second internetwork link for a value equivalent to said destination node address;

(d) retrieving from said second address mapping table, a network number of a local network and a node address of said destination node; and

(e) substituting, by said second internetwork link, said network number and said node address in place of said domain network address and said destination node address, respectively, in said packet.

10. The method of claim 9 further comprising the step of providing, by said second internetwork link, said source network number and said source node address to said first internetwork link.

11. In an internetwork link of an internetwork, a method for representing substantially all nodes in a domain as being associated with a single network number, the method comprising the steps of:

(a) receiving a packet from a first node within the domain, said packet having a network layer header with a first network layer address, said network layer address including a network number of a local network in which said first node resides, and an originally-assigned node address of said first node;

(b) translating said first network layer address to a second network layer address, wherein said second network layer address is unique within the internetwork, and wherein the step of translating further comprises the steps of:

(i) generating a domain network number of said second network layer address, wherein said domain network number is unique within the internetwork;

(ii) generating a gateway-mapped node address of said second network layer address, wherein said gateway-mapped node address is unique within the domain; and

(iii) replacing said first network layer address with said second network layer address in said network layer header of said packet;

(c) receiving a second packet, said second packet having a second network layer header with a second network layer address, said second network layer address including a network number portion having a value equivalent to said domain network number, and a node address portion having a value equivalent to said gateway-mapped node address;

(d) substituting said first network number and said node address in place of said domain network address and said gateway-mapped node address, respectively, to translate said second network layer address to said first network layer address;

(e) referencing a translation exemption table containing non-translatable network numbers; and

(f) performing said step (b) when said translation exemption table does not contain said network number of said local network.

12. The method of claim **11**, further comprising the step of:

(g) advertising said non-translatable network numbers and said domain network number into the internetwork.

13. An address mapping gateway for associating substantially all nodes in a domain with a domain network address, the address mapping gateway comprising:

(a) means for receiving a packet from a first node within the domain, said packet having a network layer header including a network layer address, said network layer address comprising a node address of said first node and a network number of a local network including said first node;

(b) means for providing a gateway-mapped node address for said first node, wherein said gateway-mapped node address is unique within the domain;

(c) means for substituting the domain network address for said network number and for substituting said gateway-mapped node address for said node address in said packet;

(d) means for receiving a second packet, destined for said first node, said second packet having a second network layer header with a second network layer address, said second network layer address including said domain network address identifying said local network including said first node and said gateway-mapped node address to identify said first node;

(e) means for substituting said network number and said node address in place of said domain network address and said gateway-mapped node address of said second packet, respectively; and

(f) an address mapping table configured to store one or more mapping entries, each said mapping entry associated with a node within the domain, said one or more mapping entries including a first node mapping entry including:

(i) said node address of said first node,

(ii) said network number of said local network in which said first node resides, and

(iii) said gateway-mapped node address;

(g) means for creating said mapping entries in said address mapping table; and

(h) a translation exemption table containing non-translatable network numbers,

wherein said address mapping gateway does not translate network layer addresses containing said non-translatable network numbers.

14. The address mapping gateway of claim **13**, wherein said address mapping gateway advertises network layer addresses having said non-translatable network numbers, and said domain network address, into the internetwork.

* * * * *

77/3,K/64 (Item 39 from file: 350) **Links**
Derwent WPIX

015254883 **Image available**
WPI Acc No: 2003-315812/200331
XRPX Acc No: N03-251503
Data mapping method for handheld wireless devices, involves producing two set of programmatic components for read configuration information and server/client information dynamically
Patent Assignee: OPENWAVE SYSTEMS INC (OPEN-N); ARBO R (ARBO-I); MIRKINA O (MIRK-I); PAMER S A (PAME-I); ROY W S (ROYW-I); ZHEGLOV A V (ZHEG-I)
Inventor: ARBO R; MIRKINA O; PAMER S A; ROY W S; ZHEGLOV A; ZHEGLOV A V
Number of Countries: 027 Number of Patents: 002
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
EP 1271360	A2	20030102	EP 2002254530	A	20020627	200331 B
US 20040093342	A1	20040513	US 2001301462	P	20010627	200432
			US 20018001	A	20011113	

Priority Applications (No Type Date): US 20018001 A 20011113; US 2001301462 P 20010627
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 1271360	A2	E	33	G06F-017/30	
Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI TR					
US 20040093342	A1			G06F-017/00	Provisional application US 2001301462

Data mapping method for handheld wireless devices, involves producing two set of programmatic components for read configuration information and server/client information dynamically

Abstract (Basic):
... A **configuration** information is read from a **data file corresponding** to a particular type of client terminals (104A-104Z) a two set of programmatic components for processing each **configuration** information and data stored in either server (101) or client terminals are produced dynamically. The data stored in client terminal and server is **mapped** by executing relevant programmatic components.
... 1) Computer program for data **mapping**; and...

...2) Data **mapping** system...

...For **mapping** personal information management data communicated between handheld wireless devices such as wireless telephone, PDA, handheld...

...Reduces implementation time and cost **associated** with information synchronization by simplifying information **mapping** of new clients **without modifying** resources and programming. Avoids unwanted **mapping** by specifying the suitable clients before **mapping** appropriately...

...The figure **shows** a block diagram of client server system...
...Title Terms: **MAP**;
International Patent Class (Main): **G06F-017/00**...

...**G06F-017/30**
Manual Codes (EPI/S-X): **T01-F05B2**...

...**T01-M06A1A**...

...**T01-N02A2C**

US 20040093342A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2004/0093342 A1

Arbo et al. (43) Pub. Date: May 13, 2004

(54) UNIVERSAL DATA MAPPING SYSTEM

(76) Inventors: Ronald Arbo, North Andover, MA (US); Olga Mirkina, Dedham, MA (US); Steven A. Pamer, Bedford, MA (US); Alexei V. Zheglov, Watervliet, NY (US); William S. Roy, Londonderry, NH (US)

Correspondence Address:
TROUTMAN SANDERS LLP
BANK OF AMERICA PLAZA, SUITE 5200
600 PEACHTREE STREET , NE
ATLANTA, GA 30308-2216 (US)

(21) Appl. No.: 10/008,001

(22) Filed: Nov. 13, 2001

Related U.S. Application Data

(60) Provisional application No. 60/301,462, filed on Jun. 27, 2001.

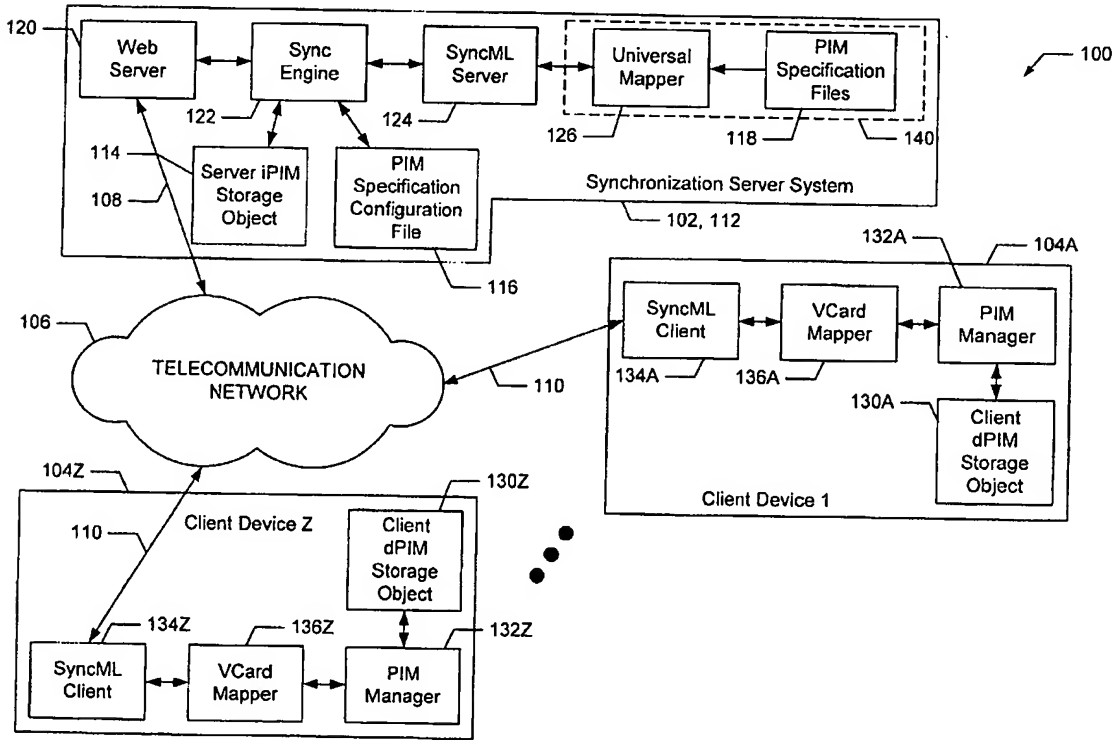
Publication Classification

(51) Int. Cl.[7] G06F 17/00

(52) U.S. Cl. 707/102

(57) ABSTRACT

A universal mapping system, including apparatuses and methods, which is configurable through use of selectable configuration files to perform the bi-directional mapping and conversion of data between different data structures and formats. Each configuration file uniquely corresponds to a particular type of device which stores mappable data in a structure and/or format different than those of other types of devices. The universal mapping system is operable to perform mappings of data according to priority rules which are definable in the configuration files and which govern the order in which the mappings of data elements are performed. The universal mapping system is also adapted to maintain associations between data elements resulting from prior mappings. Additionally, the universal mapping system is operable to persist unmappable data from a source during a mapping in one direction and to return the unmappable data to that source during a subsequent mapping in the opposite direction.



its return, to a client device **104**, in a future synchronization session which updates the client device **104**. Then, at step **920**, the universal mapper **126** transforms the mapped data to the proper format for the direction of the synchronization session (i.e., places the mapped data into a UC record object or a vCard) and returns the resulting transformed and mapped data (i.e., resultant data) to the sync engine **122** and/or SyncML server **124**, as appropriate, for updating of the server iPIM storage object **114** or communication to the client device **104**. The universal mapper **126** terminates operation in accordance with the mapping method **900** at step **922**.

[0076] The previously described methods are utilized by the synchronization server system **102** in synchronizing PIM data between a client device **104** and the synchronization server system **102** regardless, for the most part, of the synchronization direction. To aid in illustrating the results of the methods of the preferred embodiment in a sequence of exemplary synchronization sessions in which PIM data is received from a client device **104** (i.e., having a type associated with the PIM specification file **118** of **FIGS. 4A and 4B**) in the form of vCards and is used to update the server's iPIM storage object **114**, **FIG. 10A** displays a new vCard **1000** which is created at the client device **104** and which is communicated from the client device **104** to the synchronization server system **102**. Upon receipt of the new vCard **1000**, the synchronization server system **102** operates according to the methods described herein to identify the type of the client device **104** and select the appropriate PIM specification file **118** (see **FIGS. 4A and 4B**), to appropriately configure the universal mapper **126** using the selected PIM specification file **118**, to map the vCard properties to UC fields, and to produce a UC record object which, when stored by the sync engine **122**, causes the server iPIM storage object **114** to include UC fields (i.e., HomeNumber, HomeNumber2, FirstName, LastName, DisplayName, email1, and email2) having PIM data as seen in **FIG. 10B**. Note that the full text of the vCard's note property is saved by the universal mapper **126** in the UC record's history/association table for return to the client device **104** during a later synchronization session in the opposite direction.

[0077] Subsequently, the client device **104** produces a second vCard **1002** (see **FIG. 10C**) and communicates it to the synchronization server system **102**. Upon receipt of the second vCard **1002**, the synchronization server system **102** operates according to the methods described herein, in a second synchronization session, to identify the type of the client device **104** and select the appropriate PIM specification file **118** (see **FIGS. 4A and 4B**), to appropriately configure the universal mapper **126** using the selected PIM specification file **118**, to map the vCard properties to UC fields, and to produce a UC record object which, when stored by the sync engine **122**, causes the server iPIM storage object **114** to include UC fields (i.e., HomeNumber, HomeNumber2, FirstName, LastName, DisplayName, email1, and email2) having PIM data as seen in **FIG. 10D**. Note that, as a result of the receipt and processing of the second vCard **1002**, the UC record of the server iPIM storage object **114** now includes an added business telephone number and a new note which has been saved in the record's history/association table. Note, also, that a second home telephone number was removed as a result of the second synchronization session.

[0078] To aid in illustrating the results of the methods of the preferred embodiment for an exemplary synchronization session in which PIN data is retrieved from the server iPIM storage object **114** and used to update the client device's dPIM storage object **130**, **FIG. 11A** displays the server iPIM storage object **114** of **FIG. 10D** after changes have been made to add a second home telephone number, to modify the first email address, and to add a home address. After initiation of the synchronization session, the synchronization server system **102** operates in accordance with the methods described herein to identify the type of the client device **104** and select the appropriate PIM specification file **118** (see **FIGS. 4A and 4B**), to appropriately configure the universal mapper **126** using the selected PIM specification file **118**, to map the UC fields to vCard properties, and to produce the vCard **1004** depicted in **FIG. 11B**. Note that, as a result of the "MaxPerContact" configuration element **402** having a value of two, only the work telephone number and the preferred home telephone number were mapped to the vCard **1004**. Also, note that the note property of the vCard **1004** includes the note text received by the synchronization server system **102** in the second vCard **1002** (i.e., which updated the UC note field that was populated initially with the note text from the note property of the new vCard **1000**), thereby displaying the storing and retrieval of the previously unmapped note text from the history/association table by the universal mapper **126**. In addition, note that the vCard **1004** includes an address property not present in vCards **1000**, **1002**, thereby illustrating the mapping of previously unmapped PIM data by the universal mapper **126**.

[0079] It should be understood that the description of the preferred embodiment of the present invention is for descriptive or exemplary purposes only, and that the apparatuses and methods of the present invention may be utilized in a wired or wireless communications environment to map data between two data storage structures and/or formats. Also, the communications environment may include communication networks or communication connections other than the Internet and the use of protocols and specifications other than SyncML and vCard. Additionally, the present invention's apparatuses and methods may be employed in data management architectures other than client/server architectures. In addition, the apparatuses and methods of the present invention may be utilized to map any type of data other than personal information management data.

[0080] Whereas this invention has been described in detail with particular reference to its most preferred embodiment, it is understood that variations and modifications can be effected within the spirit and scope of the invention, as described herein before and as defined in the appended claims. The corresponding structures, materials, acts, and equivalents of all means or step plus function elements, if any, in the claims below are intended to include any structure, material, or acts for performing the functions in combination with other claimed elements as specifically claimed.

What is claimed is:

1. A method of mapping data comprising the steps of:

parsing a configuration file to obtain mapping configuration information;

utilizing the mapping configuration information to dynamically configure data mappers for mapping operations; and,

executing the data mappers to map data in accordance with the mapping configuration information.

* * * * *

77/3,K/44 (Item 19 from file: 350) **Links**
Derwent WPIX

016136156 **Image available**
WPI Acc No: 2004-294032/200427
XRPX Acc No: N04-233549
Shared web hosting system includes servers connected to table which stores non-changing information and dynamic mapping information for web pages
Patent Assignee: INTERLAND INC (INTE-N); COLEMAN R (COLE-I)
Inventor: COLEMAN R J; COLEMAN R
Number of Countries: 107 Number of Patents: 004
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 20040054793	A1	20040318	US 2002411214	P	20020916	200427	B
			US 2002286439	A	20021101		
WO 200425491	A1	20040325	WO 2003US28303	A	20030909	200427	
AU 2003270472	A1	20040430	AU 2003270472	A	20030909	200462	
TW 200417190	A	20040901	TW 2003125479	A	20030916	200624	

Priority Applications (No Type Date): US 2002411214 P 20020916; US 2002286439 A 20021101
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20040054793	A1		9	G06F-015/16	Provisional application US 2002411214
WO 200425491	A1	E		G06F-015/16	

Designated States (National): AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU CZ DE DK DM DZ EC EE EG ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NI NO NZ OM PG PH PL PT RO RU SC SD SE SG SK SL SY TJ TM TN TR TT TZ UA UG UZ VC VN YU ZA ZM ZW
Designated States (Regional): AT BE BG CH CY CZ DE DK EA EE ES FI FR GB GH GM GR HU IE IT KE LS LU MC MW MZ NL OA PT RO SD SE SI SK SL SZ TR TZ UG ZM ZW

AU 2003270472	A1		G06F-015/16	Based on patent WO 200425491
TW 200417190	A		H04L-012/00	

Shared web hosting system includes servers connected to table which stores non-changing information and dynamic mapping information for web pages

Abstract (Basic):
... serving web pages to client computer systems and transmitting web page request to a shared **table data structure** and a virtual host information server connected to storage device that includes dynamic **mapping** information. The **table** shared by the servers, stores **non-changing** information and dynamic **mapping information** for web **pages**.

... The figure **shows** a shared web hosting system...
...Title Terms: **TABLE**;
International Patent Class (Main): **G06F-015/16**...
International Patent Class (Additional): **G06F-015/167**
Manual Codes (EPI/S-X): **T01-J05B2**...

...**T01-N01D**...

...**T01-N02A3C**

US 20040054793A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2004/0054793 A1

Coleman (43) Pub. Date: Mar. 18, 2004

(54) SYSTEM AND METHOD FOR HIGH PERFORMANCE SHARED WEB HOSTING

(76) Inventor: Richard Coleman, Marietta, GA (US)

Correspondence Address:
HOWREY SIMON ARNOLD & WHITE
ATTORNEYS AT LAW
750 BERING DRIVE
HOUSTON, TX 77057-2198 (US)

(21) Appl. No.: 10/286,439

(22) Filed: Nov. 1, 2002

Related U.S. Application Data

(60) Provisional application No. 60/411,214, filed on Sep. 16, 2002.

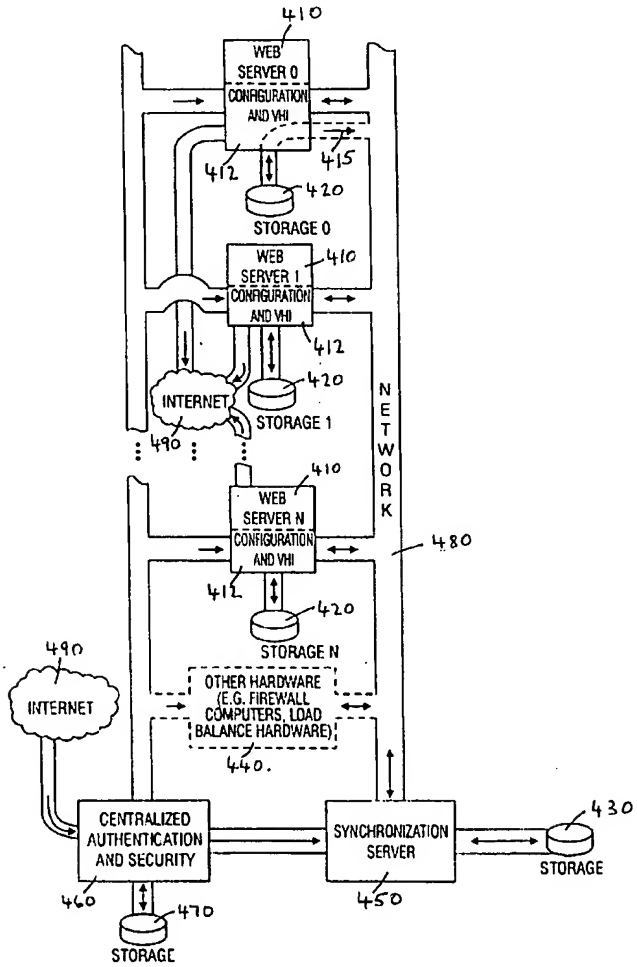
Publication Classification

(51) Int. Cl.[7] G06F 15/16; G06F 15/167

(52) U.S. Cl. 709/229; 709/218; 709/213

(57) ABSTRACT

A system for shared web hosting includes a plurality of web servers coupled to a shared table data structure, wherein the web servers serve web pages to client computer systems. The web servers all couple to a security server that transmits web page requests to the shared table data structure. A website configuration server and virtual host information server couple to the shared table. The virtual host information server couples to a storage device that includes dynamic mapping information. Dynamic mapping information identifies the web server or web servers hosting a web site at any given time. The website configuration server includes static non-changing configuration information for each owner's web site. The non-changing information describes web sites hosted on the web server. The shared table also stores copies of recently accessed non-changing information and dynamic mapping information for web pages.



practiced in different but equivalent manners apparent to those skilled in the art having the benefit of the teachings herein. It is therefore evident that the particular embodiments disclosed above may be altered or modified and all such variations are considered within the scope and spirit of the invention. It is intended that the following claims be interpreted to enhance all such variations and modifications.

References

[0027] The following references, to the extent that they provide exemplary procedural or other details supplementary to those set forth herein, are specifically incorporated herein by reference.

[0028] 1. Apache Software Foundation, "Apache HTTP Server Version 1.3-Using the Apache HTTP Server", http://httpd.apache.org/docs/, 2002

[0029] 2. Apache Software Foundation, "Apache HTTP Server Version 2.0 Documentation", http://httpd.apache.org/docs-2.0/, 2002

What is claimed is:

1. A system for shared web hosting, comprising:

at least two of a first computer means coupled to a shared table, wherein the first computer means serve web pages to clients;

a second computer means coupled to the first computer means, wherein the second computer means transmits web page requests to the shared table;

a third computer means coupled to the shared table;

a fourth computer means coupled to the shared table, wherein the fourth computer means couples to a storage device that includes dynamic mapping information; and

wherein the shared table stores non-changing information and dynamic mapping information for web pages.

2. The system of claim 1, wherein the dynamic mapping information identifies the first computer mean hosting a web site at any given time.

3. The system of claim 2, wherein the first computer mean is a web server.

4. The system of claim 1, wherein the second computer means transmits web page requests to the first computer means.

5. The system of claim 4, wherein the second computer means is a security server.

6. The system of claim 1, wherein the non-changing information for all web sites is stored on a storage device coupled to the third computer means, wherein the non-changing information describes web sites hosted on the first computer mean.

7. The system of claim 6, wherein the third computer means is a website configuration server.

8. The system of claim 1, further comprising:

one or more hardware devices executing firewall software, wherein said hardware devices couple to the first computer means; and

one or more load balancing devices coupled to the first computer means.

9. The system of claim 1, wherein the second computer means receives web page requests through the Internet from clients.

10. The system of claim 10, wherein the fourth computer means is a virtual host information server.

11. A method for shared web hosting, comprising:

verifying that a web page request is allowed access to information in a web site;

transmitting the web page request to a plurality of web servers and a control means after verification;

performing a lookup of a data structure means to determine identification information for the web page request, wherein the identification information includes website configuration information and virtual host information; and

routing the web page request to one or more of the web servers.

12. The method of claim 11, wherein the step of verifying further comprises authenticating user identification information to determine web site access.

13. The method of claim 11, wherein verification is performed by a centralized authentication server (CAS).

14. The method of claim 13, wherein verification information is stored on computer means coupled to the CAS.

15. The method of claim 11, wherein the control means is a shared table controller.

16. The method of claim 11, wherein the data structure means is a shared table.

17. A system for shared web hosting, comprising:

at least two of a first computer means coupled to a second computer means, wherein the first computer means each couple to a storage device that includes web site configuration and virtual host information;

a third computer means coupled to each of the first computer means, wherein the third computer means transmits web page requests to the second computer means; and

wherein the second computer means periodically update configuration and virtual host information on all the first computer means to maintain coherency.

18. The system of claim 17, wherein the first computer means is a web server.

19. The system of claim 17, wherein the second computer means is a synchronization server.

20. The system of claim 17, wherein the third computer means is a security server.

* * * * *

77/3,K/60 (Item 35 from file: 350) **Links**
Derwent WPIX

015340548 **Image available**
WPI Acc No: 2003-401486/200338
XRPX Acc No: N03-320179

Inter-processor communication system for parallel computer system, transmits packet requesting translation of write address before transmission of data, from source processor to destination processor

Patent Assignee: NEC CORP (NIDE)
Inventor: KANOH Y
Number of Countries: 002 Number of Patents: 002
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 20030005071	A1	20030102	US 2002161636	A	20020605	200338	B
JP 2002366427	A	20021220	JP 2001170377	A	20010606	200340	

Priority Applications (No Type Date): JP 2001170377 A 20010606
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20030005071	A1		13	G06F-015/167	
JP 2002366427	A		11	G06F-012/02	

Inter-processor communication system for parallel computer system, transmits packet requesting translation of write address before transmission of data, from source processor to destination processor

Abstract (Basic):

... A transmitter (4) transmits a packet requesting **translation** of a write **address** before transmission of data, from a **source** processor to a **destination** processor. An **address** translating circuit (54) **translates** the write **address** to a physical **address**. A **data** write circuit (57) uses the result of the **address translation** as the write **address** for writing the data transmitted subsequent to the packet.

... **Eliminates** need for **address translation** at the initial normal packet with data arrival and hence reduces time required for **address translation** in the receiver. Also reduces capacity of the memory for holding the **address translation** result and reduces overhead of the **address translation** in the receiver...

...The figure **shows** a block diagram illustrating the inter-processor communication system...

...write **address** register (53...

...**address translating** circuit (54...

...**translation table** (55
...Title Terms: **TRANSLATION;**
International Patent Class (Main): **G06F-012/02**...

...**G06F-015/167**
International Patent Class (Additional): **G06F-012/06**...

...**G06F-015/163**...

...**G06F-015/177**
Manual Codes (EPI/S-X): **T01-H07C7**...

...**T01-M02A**...

...**T01-M02C**

US 20030005071A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2003/0005071 A1

Kanoh (43) Pub. Date: Jan. 2, 2003

(54) INTER-PROCESSOR COMMUNICATION SYSTEM AND METHOD

(75) Inventor: Yasushi Kanoh, Tokyo (JP)

Correspondence Address:
FOLEY AND LARDNER
SUITE 500
3000 K STREET NW
WASHINGTON, DC 20007 (US)

(73) Assignee: NEC CORPORATION

(21) Appl. No.: 10/161,636

(22) Filed: Jun. 5, 2002

(30) Foreign Application Priority Data

Jun. 6, 2001 (JP) 2001-170377

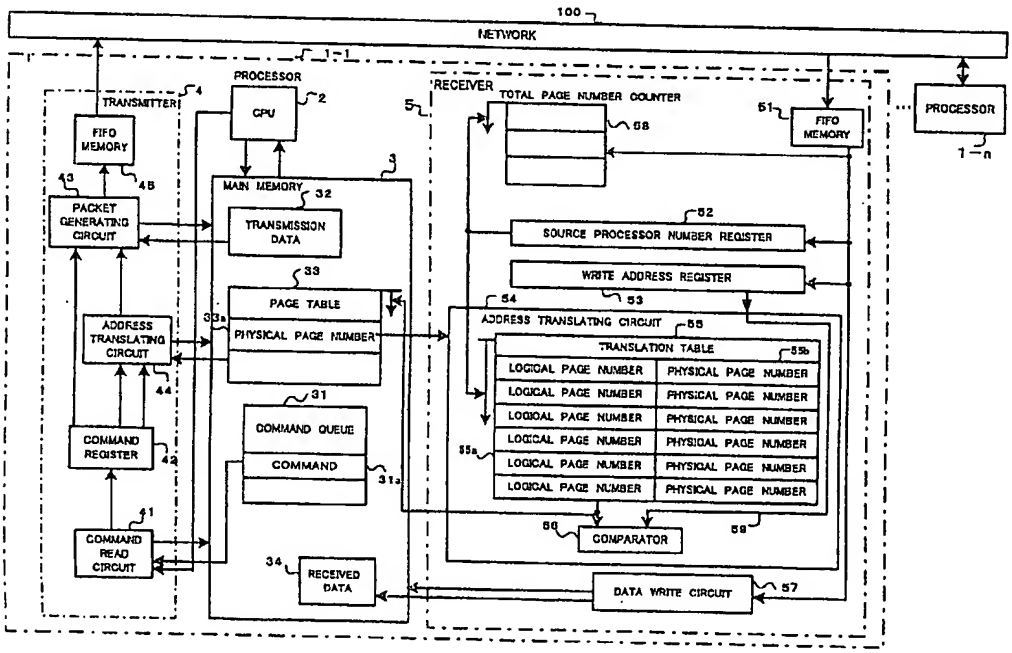
Publication Classification

(51) Int. Cl.[7] G06F 15/167

(52) U.S. Cl. 709/213

(57) ABSTRACT

A transmitter 43 transmits a write address (i.e., address translation packet before sending the first data packet of a command). A receiver 5, when receiving the address translation packet, executes address translation of a write address. The receiver 5 also preliminarily executes address translation in advance during inter-processor communication for reducing overhead of address translation in the destination processor that occurs for each page. The transmitter reports the total number of pages in advance for suppressing wasteful address translation subsequent to the last page.



What is claimed is:

1. An inter-processor communication system of parallel computer in which a plurality of processors are connected by a network, each said processor including:

a transmitting means for transmitting a packet for requesting address translation of a write address in a destination processor as designated by logical address to the destination processor when transmitting data thereto;

a means for translating said write address transmitted from the source processor to a physical address; and

a means for using the result of the address translation as a write address for writing data transmitted subsequent to said packet for requesting the address translation.

2. The inter-processor communication system according to claim 1, wherein said transmitting means transmits said packet for requesting address translation of the write address to the destination processor before translating the read address of the transmission data as designated by the logical address to physical address.

3. The inter-processor communication system according to claim 2, wherein a write address is designated in the packet for transmitting the transmission data therein, and when a write address added to the previously transmitted said packet for requesting the address translation and the write address designated by said packet for transmitting the transmission data therein are different, the write address designated in said packet for transmitting the transmission data therein is re-translated for use.

4. An inter-processor communication system of parallel computer in which a plurality of processors are connected by a network, each processor including:

a means for translating a write address in a destination processor as designated by logical address to physical address at the time of packet reception in the destination processor;

a means for storing the result of the write address translation; and

a means for executing subsequently necessary write address translation in advance before the write address translation result stored earlier becomes incapable of being used.

5. An inter-processor communication system of parallel computer in which a plurality of processors are connected by a network, each processor including:

a means for translating a read address of transmission data designated by logical address to physical address when transmitting data to the destination processor; and

a means for storing the result of the read address translation, and a means for executing subsequently necessary read address translation in advance before the read address translation result stored earlier becomes incapable of being used.

6. An inter-processor communication system of parallel computer in which a plurality of processors are connected by a network, each processor including:

a means for translating a write address in a destination processor as designated by logical address to physical address at the time of packet reception in the designation processor;

a means for storing the result of the write address translation;

a means for executing subsequently necessary write address translation in advance before the write address translation result stored earlier becomes incapable of being used;

a means for translating a read address of transmission data designated by logical address to physical address at the time of data transmission to the destination processor;

a means for storing the result of the read address translation; and

a means for executing subsequently necessary read address translation before the stored read address translation result becomes incapable of being used,

wherein the write address is designated in the packet for transmitting the transmission data therein, and when a write address added to the previously transmitted packet for requesting the address translation and the write address designated by the packet for transmitting the transmission data therein are different, the write address designated in the packet for transmitting the transmission data therein is re-translated for use.

7. The inter-processor communication system according to claim 6, wherein when data are transmitted in a plurality of packets under control of a communication command, said packet for requesting the address translation is transmitted only right before the first packet for transmitting the transmission data pertaining to the command, and subsequently only the packets for transmitting the transmission data are transmitted.

8. The inter-processor communication system according to claim 7, wherein said storing means for storing the write address translation result manages the write address translation result for each source processor, and stores two write address translation results, i.e., one being in use and the other one obtained by in-advance address translation, for the same source processor.

9. The inter-processor communication system according to claim 8, wherein said packet for requesting the address translation has the total number of pages in the destination processor of data transmitted under control of the command, and the destination processor counts down the number of pages in order to predict the last page to be written under control of the command, thus suppressing wasteful in-advance address translation subsequent to the last page.

10. An inter-processor communication method of parallel computer in which a plurality of processors are connected by a network, each said processor including steps of:

a step for transmitting a packet for requesting address translation of a write address in a destination processor as designated by logical address to the destination processor when transmitting data thereto;

a step for translating said write address transmitted from the source processor to a physical address; and

a step for using the result of the address translation as a write address for writing data transmitted subsequent to said packet for requesting the address translation.

11. The inter-processor communication method according to claim 10, wherein said transmitting step transmits said packet for requesting address translation of the write address

to the destination processor before translating the read address of the transmission data as designated by the logical address to physical address.

12. The inter-processor communication method according to claim 11, wherein the write address is designated in the packet for transmitting the transmission data therein, and when a write address added to the previously transmitted said packet for requesting the address translation and the write address designated by said packet for transmitting the transmission data therein are different, the write address designated in said packet for transmitting the transmission data therein is re-translated for use.

13. An inter-processor communication method of parallel computer in which a plurality of processors are connected by a network, each processor including steps of:

a step for translating a write address in a destination processor as designated by logical address to physical address at the time of packet reception in the destination processor;

a step for storing the result of the write address translation; and

a step for executing subsequently necessary write address translation in advance before the write address translation result stored earlier becomes in capable of being used.

14. An inter-processor communication method of parallel computer in which a plurality of processors are connected by a network, each processor including steps of:

a step for translating a read address of transmission data designated by logical address to physical address when transmitting data to the destination processor; and

a step for storing the result of the read address translation, and a means for executing subsequently necessary read address translation in advance before the read address translation result stored earlier becomes incapable of being used.

15. An inter-processor communication method of parallel computer in which a plurality of processors are connected by a network, each processor including steps of:

a step for translating a write address in a destination processor as designated by logical address to physical address at the time of packet reception in the designation processor;

a step for storing the result of the write address translation;

a step for executing subsequently necessary write address translation in advance before the write address translation result stored earlier becomes in capable of being used;

a step for translating a read address of transmission data designated by logical address to physical address at the time of data transmission to the destination processor;

a step for storing the result of the read address translation; and

a step for executing subsequently necessary read address translation before the stored read address translation result becomes incapable of being used,

wherein the write address is designated in the packet for transmitting the transmission data therein, and when a write address added to the previously transmitted packet for requesting the address translation and the write address designated by the packet for transmitting the transmission data therein are different, the write address designated in the packet for transmitting the transmission data therein is re-translated for use.

16. The inter-processor communication method according to claim 15, wherein when data are transmitted in a plurality of packets under control of a communication command, said packet for requesting the address translation is transmitted only right before the first packet for transmitting the transmission data pertaining to the command, and subsequently only the packets for transmitting the transmission data are transmitted.

17. The inter-processor communication method according to claim 16, wherein said storing step for storing the write address translation result manages the write address translation result for each source processor, and stores two write address translation results, i.e., one being in use and the other one obtained by in-advance address translation, for the same source processor.

18. The inter-processor communication method according to claim 17, wherein said packet for requesting the address translation has the total number of pages in the destination processor of data transmitted under control of the command, and the destination processor counts down the number of pages in order to predict the last page to be written under control of the command, thus suppressing wasteful in-advance address translation subsequent to the last page.

* * * * *

77/3,K/36 (Item 11 from file: 350) **Links**
Derwent WPIX

016623225 **Image available**
WPI Acc No: 2004-781951/200477
XRPX Acc No: N04-615980
Round-robin arbiter for network device e.g. router, has unmapping unit that converts selected data item modified based on randomization values, into original data item based on prestored corresponding de-randomization value

Patent Assignee: JUNIPER NETWORKS INC (JUNI-N)
Inventor: GUPTA A P; LACROUTE P; SINDHU P; ZHANG S
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 6807594	B1	20041019	US 2001905071	A	20010716	200477	B

Priority Applications (No Type Date): US 2001905071 A 20010716
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 6807594	B1		13	G06F-013/14	

Round-robin arbiter for network device e.g. router, has unmapping unit that converts selected data item modified based on randomization values, into original data item based on prestored corresponding de-randomization value

Abstract (Basic):
... A **mapping** unit **modifies** input data items, based on randomization values stored in a randomization **table** (401). An arbiter unit (405) selects one of the **modified** data items, based on round-robin arbitration scheme. An **un**-mapping unit (421) converts the selected data item to **original** data item based on **corresponding** de-randomization values stored in a de-randomization **table** (402).
... Round-round arbiter (RRA) for network device (claimed) such as **router**.
...

...The figure **shows** a block diagram of the round-round arbiter...

...randomization **table** (401...

...de-randomization **table** (402...
...un-**mapping** unit (421
...Title Terms: **ROUTER**;

International Patent Class (Main): **G06F-013/14**
Manual Codes (EPI/S-X): **T01-E04**...

...**T01-F02C2**...

...**T01-H05B**...

...**T01-N02A3B**

US006807594B1

(12) **United States Patent**
Sindhu et al.

(10) **Patent No.: US 6,807,594 B1**
(45) **Date of Patent: Oct. 19, 2004**

(54) **RANDOMIZED ARBITERS FOR ELIMINATING CONGESTION**

(75) Inventors: **Pradeep Sindhu**, Los Altos Hill, CA (US); **Song Zhang**, San Jose, CA (US); **Anurag P. Gupta**, Saratoga, CA (US); **Phil Lacroute**, Sunnyvale, CA (US)

(73) Assignee: **Juniper Networks, Inc.**, Sunnyvale, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 651 days.

(21) Appl. No.: **09/905,071**

(22) Filed: **Jul. 16, 2001**

(51) **Int. Cl.**[7] **G06F 13/14**; H04L 12/56; H04L 12/28

(52) **U.S. Cl.** **710/240**; 370/395.31

(58) **Field of Search** 710/107–125, 710/36–45, 240–244; 370/351, 395.3–395.42; 709/238–244

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,267,235	A *	11/1993	Thacker	370/396
6,343,066	B2 *	1/2002	Magill et al.	370/230
2003/0081548	A1 *	5/2003	Langevin et al.	370/230

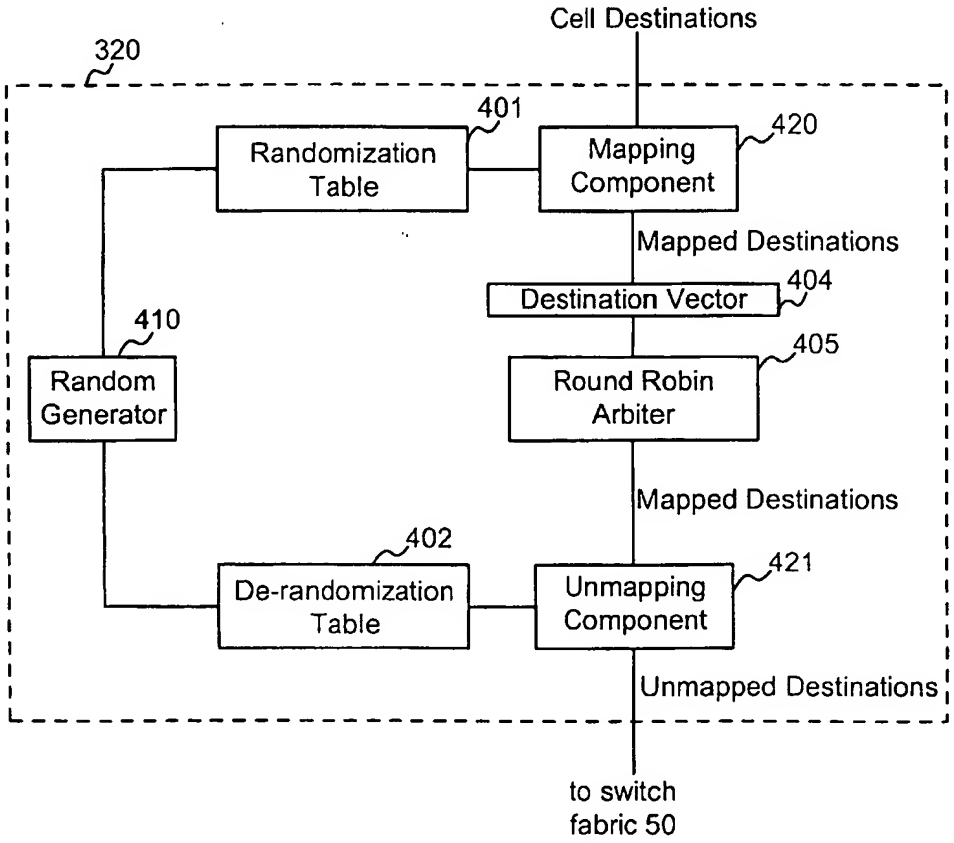
* cited by examiner

Primary Examiner—Sumati Lefkowitz
(74) *Attorney, Agent, or Firm*—Harrity & Snyder LLP

(57) **ABSTRACT**

A system having multiple arbiters is constructed to reduce the chances of arbiters synchronizing with one another. Each arbiter includes a random process that introduces randomness into an arbitration scheme performed by the arbiter. Because of the randomness, the arbiters will not tend to synchronize with one another, even when receiving an identical stream of input values.

30 Claims, 7 Drawing Sheets



destination vector **704**. Randomization arbiter **705** is based on a modified round robin arbiter **706** that periodically receives a pointer to a starting destination within destination vector **704** from a random generator, such as linear feedback shift register (LFSR) **707**. LFSRs can produce pseudo-random numbers at very high frequencies. Hardware constructions for LFSRs are well known in the art and will not be described further herein. Other circuits for generating random numbers, in addition to LFSRs, may alternatively be used.

In operation, round robin arbiter **706** receives a pointer value from LFSR **707** that indicates at which destination address in destination vector **704** the round robin arbiter should begin. If, for example, the pointer value is ten, round robin arbiter **705** begins by checking the 10^{th} bit in destination vector **704**. The round robin arbiter will then sequentially check the bits in the destination vector **704** up through the 143^{rd} bit, and then wrap around to bit zero and sequentially check the bits in destination vector **704** up through bit nine. After bit nine, round robin arbiter **706** gets a new pointer value from LFSR **707**, and beginning at the new value, checks each of the bits in destination vector **704**. The new pointer value generated by LFSR **707** may be generated periodically or in response to a new pointer request signal from round robin arbiter **706**.

Arbiter **720** provides similar functionality as the arbiter shown in FIG. **4**, due to the fact that both arbiters introduce an element of randomness into their selections. More particularly, multiple randomization arbiters **720** in a system receive different LSFR initial seed values, and will thus not have a tendency to become in-stride with one another.

Although described in the context of a purpose-built router, concepts consistent with the present invention can be implemented in any system that uses multiple arbiters where it is desirable to keep two or more of the arbiters from becoming in-stride with one another. Although the arbiters described herein arbitrate across destinations, an arbiter consistent with the principles of the invention may be used in any system requiring arbitration. For example, data items other than destinations may be randomized.

The foregoing description of preferred embodiments of the present invention provides illustration and description, but is not intended to be exhaustive or to limit the invention to the precise form disclosed. Modifications and variations are possible in light of the above teachings or may be acquired from practice of the invention.

The scope of the invention is defined by the claims and their equivalents.

What is claimed is:

1. An arbiter comprising:

a randomization table storing randomization values;

a mapping component coupled to the randomization table, the mapping component modifying data items based on the randomization values;

an arbiter component configured to select certain ones of the modified data items based on an arbitration scheme;

a de-randomization table storing de-randomization values based on the randomization values stored in the randomization table; and

an unmapping component coupled to the arbiter component and configured to convert the selected modified data items from the arbiter component back to original data items based on the de-randomization values.

2. The arbiter of claim **1**, wherein the data items are destinations in a network router.

3. The arbiter of claim **1**, further comprising:

a random generator coupled to the randomization table and the de-randomization table, the random generator generating and storing the randomization values in the randomization table and the de-randomization values in the de-randomization table.

4. The arbiter of claim **1**, further comprising:

a data item vector register coupled to the mapping component and the arbiter component, the data item vector register including a plurality of bits, each bit representing a data item.

5. The arbiter of claim **1**, wherein the arbitration scheme is a round robin arbitration scheme.

6. An arbiter comprising:

a register including a plurality of bits, each bit representing a data item, and each bit being set when information associated with the data item is waiting to be transmitted;

a random generator configured to generate a random pointer that relates to one of the bits of the register; and

an arbiter configured to select data items by checking the plurality of bits of the register including the bit pointed to by the random generator.

7. The arbiter of claim **6**, wherein the random generator generates a new random pointer for the arbiter after the round robin arbiter completes a cycle of checking the bits of the register.

8. The arbiter of claim **7**, wherein the random generator is a linear feedback shift register.

9. The arbiter of claim **6**, wherein the arbiter, after checking one of the bits in the register, clears the bit.

10. A method of arbitrating among multiple data items comprising:

randomizing the multiple data items;

selecting certain ones of the data items in accordance with an arbitration scheme based on the randomized data items; and

de-randomizing the selected data items.

11. The method of claim **10**, wherein the arbitration scheme is a round robin arbitration scheme.

12. The method of claim **10**, wherein the data items are destination addresses.

13. The method of claim **12**, further comprising:

transmitting information to destinations based on the selected data items.

14. The method of claim **10**, further comprising:

randomizing the data items by looking up a randomized value in a randomization table.

15. The method of claim **10**, further comprising:

de-randomizing the selected data items by looking up an original value associated with the data item in a de-randomization table.

16. A system comprising:

a first arbiter receiving a first set of N inputs, where N is an integer, the first arbiter selecting ones of the first set of inputs based on an arbitration scheme that includes an element of randomness when selecting the ones of the first set of inputs; and

a second arbiter receiving a second set of N inputs, the second arbiter selecting ones of the second set of inputs based on an arbitration scheme that includes an element of randomness when selecting the ones of the second set of inputs, the elements of randomness in the first and second arbiters preventing the first and second arbiters from synchronizing with one another.

17. The system of claim 16, wherein the first and second sets of N inputs are destinations associated with information to be transmitted over a network.

18. The system of claim 17, wherein the network is a fabric switch of a router.

19. The system of claim 16, wherein the arbitration scheme is a round robin arbitration scheme.

20. The system of claim 16, wherein each of the first and second arbiters further comprise:

a randomization table for storing randomization values that introduce the randomness into the arbitration scheme when selecting the ones of the first and second set of inputs; and

a mapping component for converting information in the received first and second set of inputs to a randomized version of the information based on the randomization values stored in the randomization table.

21. The system of claim 20, wherein each of the first and second arbiters further comprise:

a random generator coupled to the randomization table and configured to generate the randomization values stored in the randomization table.

22. The system of claim 20, wherein each of the first and second arbiters further comprise:

a de-randomization table for storing de-randomization values for removing the randomness introduced by the randomization values and the mapping component; and

an unmapping component configured to convert the randomized version of the information back to an original version of the information based on the de-randomization values.

23. The system of claim 16, wherein each of the first and second arbiters further comprise:

a register for storing destinations of the received N inputs, the register including a plurality of bits, each bit representing a possible destination for the N inputs, and each bit being set when an input associated with the destination is waiting to be transmitted to the network;

a random generator configured to generate a random pointer that relates to one of the bits of the register; and

a round robin arbiter configured to select an input by sequentially checking each of the plurality of bits of the register, the round robin arbiter, after completing the checking of each of the plurality of bits of the register, repeating the sequentially checking each of the plurality of bits of the register beginning at an initial location determined by the random pointer.

24. A network device comprising:

means for randomizing information associated with an input stream of data items;

means for arbitrating among the randomized information to select certain ones of the data items; and

means for de-randomizing the selected information.

25. The network device of claim 24, further comprising:

a randomization table for storing randomization values that indicate how information associated with the data items is to be modified; and

mapping means for modifying the information based on the values stored in the randomization table.

26. The network device of claim 25, further comprising:

a de-randomization table for storing de-randomization values based on the randomization values stored in the randomization table; and

unmapping means for converting the selected modified information from the mapping means back to original versions of the data items based on the de-randomization values.

27. A method comprising:

selecting ones of a first set of N inputs, where N is an integer, based on an arbitration scheme that includes an element of randomness when selecting the ones of the first set of inputs; and

selecting ones of a second set of N inputs, where N is an integer, based on an arbitration scheme that includes an element of randomness when selecting the ones of the second set of inputs, the elements of randomness in the first and second selections preventing the first and second selections from synchronizing with one another.

28. The method of claim 27, wherein the first and second sets of N inputs are destinations associated with information to be transmitted over a network.

29. The method of claim 28, wherein the network is a fabric switch of a router.

30. The method of claim 27, wherein the arbitration scheme is a round robin arbitration scheme.

* * * * *

77/3,K/32 (Item 7 from file: 350) **Links**
Derwent WPIX

017019680 **Image available**
WPI Acc No: 2005-343997/200535
Related WPI Acc No: 2005-293162
XRPX Acc No: N05-281000

Subscriber access controlling method for use in virtual private network, involves authorizing subscriber to access service on communication network, where authorization is based on domain configuration override attribute

Patent Assignee: CISCO TECHNOLOGY INC (CISC-N)
Inventor: BURNS G D; SHETH P A; SITARAMAN A; YAGER C T
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 20050086495	A1	20050421	US 2000712780	A	20001113	200535 B
			US 2004973550	A	20041025	

Priority Applications (No Type Date): US 2000712780 A 20001113; US 2004973550 A 20041025
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 20050086495	A1		11	H04K-001/00	Cont of application US 2000712780

... **network, involves authorizing subscriber to access service on communication network, where authorization is based on domain configuration override attribute**

Abstract (Basic):
... The method involves **receiving** a communication from a subscriber using a communication network coupled to **another** communication network. The communication includes a **domain** identifier **associated** with a service on the latter network. A subscriber is authorized to access a service on the latter network using one of virtual circuits. The authorizing is based on a **domain configuration** override attribute **associated** with the circuit.
... method prevents unauthorized point-to-point sessions from being forwarded to a destination layer 2 **tunneling** protocol network server and **eliminates** the need to **alter** the original point-to-point authentication packet...

...The drawing **shows** a differentiated computer network...

...**Domain** (320
...Title Terms: **DOMAIN**;

US 20050086495A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2005/0086495 A1

Sheth et al. (43) Pub. Date: Apr. 21, 2005

(54) PPP DOMAIN NAME AND L2TP TUNNEL SELECTION CONFIGURATION OVERRIDE

(75) Inventors: Purnam Anil Sheth, Livermore, CA (US); Aravind Sitaraman, Bangalore (IN); Charles T. Yager, Cupertino, CA (US); Gregory D. Burns, Pleasanton, CA (US)

Correspondence Address:
David B. Ritchie
Thelen Reid & Priest LLP
P.O. Box 640640
San Jose, CA 95164-0640 (US)

(73) Assignee: CISCO TECHNOLOGY, INC.

(21) Appl. No.: 10/973,550

(22) Filed: Oct. 25, 2004

Related U.S. Application Data

(63) Continuation of application No. 09/712,780, filed on Nov. 13, 2000.

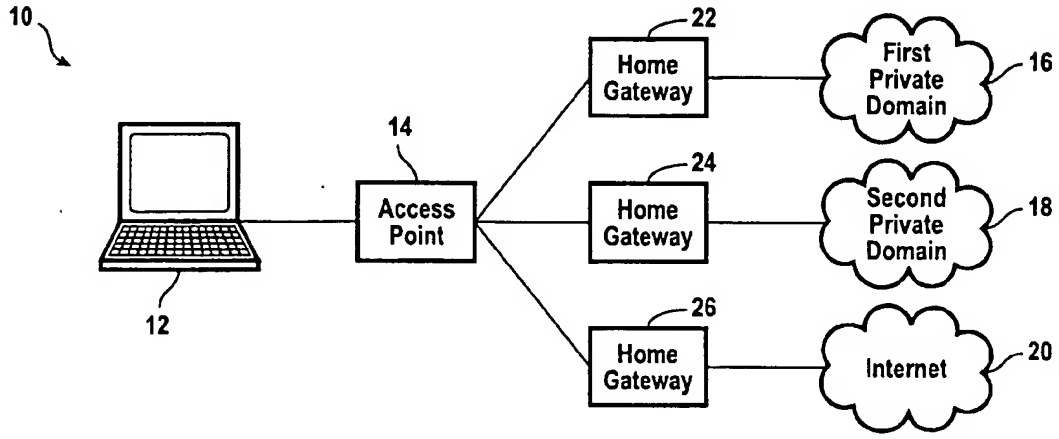
Publication Classification

(51) Int. Cl.[7] H04K 1/00

(52) U.S. Cl. 713/182

(57) ABSTRACT

A method for controlling subscriber access in a network capable of establishing connections with multiple services includes receiving a communication from a subscriber using a first communication network coupled to a second communication network, the communication optionally including a domain identifier associated with a service on the second communication network, and authorizing the subscriber to access a service on the second communication network using a virtual circuit. The authorization is based upon a domain configuration override attribute associated with the virtual circuit used to receive the communication from the subscriber. An access server capable of forcing subscribers of a communications system to gain access exclusively to a domain network associated with a virtual circuit includes an authorizer to grant service authorization to the subscribers based upon a virtual circuit used to make a service request, a virtual circuit profile request generator to generate virtual circuit profile requests and a calculator to determine whether the service associated with the virtual circuit matches the service associated with a domain configuration override attribute.



configuration override attribute exists or if the PPP authentication packet does not include a domain name (710), at 715, the domain associated with the virtual channel is returned. If a domain configuration override attribute does not exist; at 720, the PPP domain used in the PPP authentication request is returned.

[0048] Turning now to FIG. 8, a flow diagram that illustrates a method for determining the tunnel ID associated with a virtual circuit in accordance with one embodiment of the present invention is presented. At 800, a PPP session including a virtual channel ID is received. At 805, a determination is made regarding whether a domain configuration override attribute exists in a virtual circuit profile associated with the DSLAM port used to receive the PPP session. If a domain configuration override attribute exists, at 815, the tunnel ID associated with the virtual channel is returned. If a domain configuration override attribute does not exist, at 820, the PPP domain used in the PPP authentication request is returned.

[0049] FIGS. 9A-9C are tables that illustrate tunnel configuration information that may be stored in a LAC, an AAA server, or other similar devices in accordance with embodiments of the present invention. FIG. 9A is a virtual circuit profile table that contains a list of domain names 900 indexed by virtual circuit IDs 905. A domain configuration override attribute 910 determines whether a subscriber is limited to establishing a tunnel with a particular domain.

[0050] FIG. 9B is a table that includes a list of tunnel IDs 915 indexed by domain names 920. Table 9B may be used in conjunction with table 9A to obtain a tunnel ID 915 associated with a virtual circuit ID 905.

[0051] FIG. 9C is a virtual circuit profile table that contains a list of tunnel IDs 925 indexed by virtual circuit IDs 930. A domain configuration override attribute 935 determines whether a subscriber is limited to establishing a tunnel with a particular domain. In the example, a port having a virtual circuit ID of 94/22 (940) may use tunnel ID 2210 (945) exclusively.

[0052] The tunnel selection configuration override attribute is requested by the domain owner to be placed in virtual circuit profiles. It allows the service provider the capability to ensure that a PPP session originating from a DSLAM port allocated to a particular domain can connect with only that particular domain, regardless of what domain name is entered in the PPP authentication packet. This provides added security to the owner of the private domain by lessening the likelihood of an unauthorized access to the home gateway of a corporate intranet. The service provider would have the control over which ports are allocated to which domains. The service provider would also have control over which ports have the tunnel selection configuration attribute in their virtual circuit profile and are, thus, limited to one domain and which virtual circuit profiles do not contain the tunnel selection configuration override attribute and are, thus, free to connect to more than one domain.

[0053] Although embodiments of the present invention is have been described with respect to virtual circuits in an ATM networking environment, it should be understood that a virtual circuit assigned to a subscriber in system may be defined in any suitable networking environment using any suitable communication technologies and protocols, without deviating from the scope of the present invention.

[0054] In accordance with a specific embodiment of the present invention, the components, process steps, and/or data structures are implemented using software. This implementation is not intended to be limiting in any way. Different implementations may be used and may include other types of operating systems, computing platforms, and/or computer programs. In addition, those of ordinary skill in the art will readily recognize that devices of a less general purpose nature, such as hardwired devices, devices relying on FPGA (field programmable gate array) or ASIC (application specific integrated circuit) technology, or the like, may also be used without departing from the scope and spirit of the inventive concepts disclosed herewith.

[0055] While embodiments and applications of this invention have been shown and described, it would be apparent to those skilled in the art having the benefit of this disclosure that many more modifications than mentioned above are possible without departing from the inventive concepts herein. The invention, therefore, is not to be restricted except in the spirit of the appended claims.

1. A method for controlling subscriber access in a network capable of establishing connections with a plurality of services, comprising:

receiving a communication from a subscriber using a first communication network coupled to a second communication network, said communication optionally including a domain identifier associated with a service on said second communication network; and

authorizing said subscriber to access a service on said second communication network using one of a plurality of virtual circuits, said authorizing based upon a domain configuration override attribute associated with the virtual circuit used to receive said communication from said subscriber.

2-36. (canceled)

* * * * *

77/3,K/27 (Item 2 from file: 350) **Links**
Derwent WPIX

017726601 **Image available**
WPI Acc No: 2006-237878/200625
XRPX Acc No: N06-203977

Data processing method for coordinating changes made in server side data representation, involves creating modifiable working copy including components of another modifiable working copy corresponding to matched components

Patent Assignee: UNISYS CORP (BURS)
Inventor: DAHM D M; JANZIG R C
Number of Countries: 001 Number of Patents: 001
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week	
US 7016906	B1	20060321	US 2000564247	A	20000504	200625	B

Priority Applications (No Type Date): US 2000564247 A 20000504
Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 7016906	B1		27	G06F-017/00	

Data processing method for coordinating changes made in server side data representation, involves creating modifiable working copy including components of another modifiable working copy corresponding to matched components

Abstract (Basic):
... The **database** description of data source at server is **altered**, in response to administrator **changes** entered at server. A dual Schema File is created at client, by creating **unmodifiable** copy **corresponding** to the **altered** **database** description. A **modifiable** working copy including components of **another modifiable** working copy **corresponding** to **matched** components found from comparing **unmodifiable** copies, is created.
... For coordinating **changes** made in server side **database** representation with user **modifications** made to dual schema file maintained on client-side workstation...

...Enables preserving user **modifications** to the **modifiable** copy as much as possible, by performing **matching** process on the client side, when the server side data base description is **altered**.
...
...The figure **shows** the flowchart of the data processing method

...Title Terms: **CHANGE**;
International Patent Class (Main): **G06F-017/00**
Manual Codes (EPI/S-X): **T01-F05E**...

...**T01-G03**...

...**T01-H01C3**...

...**T01-J05B4P**...

...**T01-N02A2C**...

...**T01-N02A3C**...

...**T01-S03**

US007016906B1

(12) **United States Patent**
Janzig et al.

(10) **Patent No.: US 7,016,906 B1**
(45) **Date of Patent: Mar. 21, 2006**

(54) **DATA PROCESSING METHOD AND APPARATUS EMPLOYING OLE DB AND HAVING DUAL SCHEMA AND AUTO UPDATE FEATURES**

(75) Inventors: **Richard Charles Janzig**, Mission Viejo, CA (US); **David Michael Dahm**, Mission Viejo, CA (US)

(73) Assignee: **Unisys Corporation**, Blue Bell, PA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/564,247**

(22) Filed: **May 4, 2000**

(51) **Int. Cl.**
G06F 17/00 (2006.01)

(52) **U.S. Cl.** ..... **707/101**; 707/102; 707/103 R; 707/201; 707/205

(58) **Field of Classification Search** 707/100, 707/101, 102, 103, 201, 202, 203, 204, 205
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,600,778 A * 2/1997 Swanson et al. 345/762
5,758,337 A * 5/1998 Hammond 707/6
5,926,816 A * 7/1999 Bauer et al. 707/201
6,085,198 A * 7/2000 Skinner et al. 707/101
6,119,130 A * 9/2000 Nguyen et al. 707/100
6,212,534 B1 * 4/2001 Lo et al. 707/511
6,216,136 B1 * 4/2001 Ronstrom 707/203
6,401,239 B1 * 6/2002 Miron 707/203
6,453,310 B1 * 9/2002 Zander 707/1

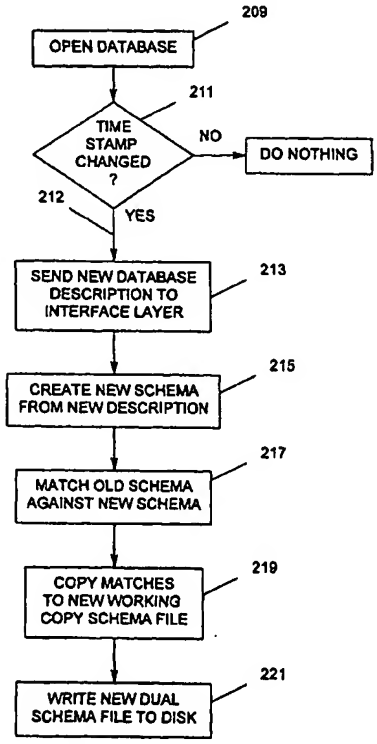
* cited by examiner

Primary Examiner—Jean M. Corrielus
Assistant Examiner—Isaac M. Woo
(74) *Attorney, Agent, or Firm*—Mark T. Starr; Lise A. Rode; Beth L. McMahon

(57) **ABSTRACT**

Mapping of data source schema data types relating a server side database description into a set of OLE DB data types is rendered subject to user modification by provision of a dual Schema File at a client side terminal which contains an original unmodifiable copy and a modifiable working copy of the data source schema. When the server side database description is altered, user modifications to the modifiable copy are preserved as much as possible by performing a matching process on the client side in the course of generating new unmodifiable and modifiable schema files which comprise a new dual Schema File.

17 Claims, 19 Drawing Sheets



copy of the data source schema, the data source schema corresponding to a database description of a data source at a server;

altering the database description at the server;

transmitting the altered database description to the client;

receiving the altered database description at the client; and

creating a second dual Schema File at the client to replace the first dual Schema File, the second dual Schema File comprising a second unmodifiable copy and a second modifiable copy, the operation of creating the second dual Schema File comprising:

creating the second unmodifiable copy corresponding to the received altered database description;

comparing the second unmodifiable copy to the first unmodifiable copy to find matches between respective components thereof; and

creating the second modifiable working copy, the second modifiable working copy includes components of the first modifiable working copy corresponding to the matched components found from the operation of comparing.

11. The method of claim **10** wherein the operation of creating the second modifiable working copy comprises:

copying the components of the first modifiable working copy corresponding to the matched components into the second modifiable working copy.

12. An apparatus in a system wherein a data source schema describes structures and data items, with their data types, the data source schema corresponding to a database description of a data source at a server, the structures and data types having been mapped into a set of OLE DB structures and data types recognized by a client, the client being arranged to access the data source via an OLE DB data provider, the apparatus comprising:

means for maintaining a first dual Schema File at a client, the first dual Schema File comprising a first unmodifiable copy of the data source schema and a first modifiable working copy of the data source schema;

means for altering the database description of the data source at the server;

means for transmitting the altered database description to the client;

means for receiving the altered database description at the client; and

means for creating a second dual Schema File at the client to replace the first dual Schema File, the second dual Schema File comprising a second unmodifiable copy and a second modifiable copy, the means for creating the second dual Schema File comprising:

means for creating the second unmodifiable copy corresponding to the received altered database description;

means for comparing the second unmodifiable copy to the first unmodifiable copy to find matches between respective components thereof; and

means for creating the second modifiable working copy, the second modifiable working copy includes components of the first modifiable working copy corresponding to the matched components found from the operation of comparing.

13. The apparatus of claim **12** wherein the database description is altered in response to administrator changes entered at the server.

14. The apparatus of claim **12** wherein the means for creating the second modifiable working copy comprises:

means for copying the components of the first modifiable working copy corresponding to the matched components into the second modifiable working copy.

15. The apparatus of claim **14** further including means for checking a time stamp prior to transmitting the altered database description to the client.

16. The apparatus of claim **12** further including means for writing the second unmodifiable copy and the second modifiable working copy to disk to replace the first dual Schema File with the second dual Schema File.

17. The apparatus of claim **12** further including means for checking a time stamp prior to transmitting the altered database description to the client.

* * * * *

Set Items Description
S1 1364345 S MAP? ? OR MAPP???
S2 967506 S GUI? ? OR (GRAPHIC? OR VISUAL?)(2W)(INTERFACE? OR APP? ? OR APPLICAT?) OR DISPLAY? OR MENU? ? OR WEBPAGE?
S3 72007 S COMPUTER?()(SCREEN? OR WINDOW?) OR COMPUTER?(2N)(IMPLEMENT? OR TOOL? ? OR UTILIT?)
S4 7050 S USERINTERFAC? OR (DROP OR PULL)()DOWN? OR (WINDOW OR DESKTOP)()MANAGER? OR PULLDOWN?
S5 32847 S UI OR UIS OR (GRAPHIC? OR VISUAL? OR INFORMATION? OR DATA?)(2N)(MENU? OR PAGE? OR SCREEN?)
S6 3848928 S SOFTWARE()OBJECT? OR DATABASE? OR DOMAIN? OR COLUMN? OR TABLE? OR DATAOBJECT? OR DATASTRUCTUR? OR DATAVALUE?
S7 2895699 S OBJECT? OR DATAOBJECT? OR DATA()(ENTIT? OR STRUCTUR?) OR OBJECTMODEL? OR OBJECT()MODEL?
S8 5655 S DATACOMMAND? OR DATAORDER? OR DATAUNIT? OR DATA()UNIT? OR DATA()MESSAG? OR DATA()INSTRUCT?
S9 49849 S DATABYTE? OR DATABLOCK? OR DATA()(BLOCK? OR REQUEST? OR INPUT? OR TYPE? OR MEMBER? OR VALUE?) OR DATATYPE?
S10 60255 S (DATA? OR SOFTWARE? OR OBJECT?)(2N)(IDENTIFIER? OR TAG? ? OR METADATA? OR ADDRESS? OR LOCATER? OR LABEL? OR NAME? OR EXTENSION? OR DESIGNATION?)
S11 527101 S DATA()(REPRESENTATION? OR DEPOSITOR? OR REPOSITOR? OR BANK? OR FILE? OR COLLECTION?) OR ROW OR ROWS
S12 7880125 S LOG OR LOGS OR CATALOG? OR CHRONICL? OR RECORD? ? OR LIST??? OR FIELD? OR ARRAY? OR ARCHIV?
S13 19322 S S1 AND S2:S5 AND S6:S12
S14 4470 S MODULAT? OR CONVERT? OR MODIF? OR ALTER? OR TRANSFORM?
S15 3551 S CHANGE? OR CHANGING? OR TRANSLAT? OR AMEND? OR EDIT? OR UPDAT?
S16 1626 S CONVERSION? OR MANIPULAT? OR CUSTOMIZ? OR PERSONALIZ? OR INDIVIDUALIZ?
S17 1729 S CUSTOMIS? OR ADAPT? OR RECONFIGUR? OR CONFIGUR? OR REVIS? OR REARRANG?
S18 10 S PERSONALIS? OR (CUSTOM OR TAILOR)()(MAKE? OR MADE? OR MAKING)
S19 55 S REORDER? OR RESHUFFL? OR RESET? OR REFORMAT? OR READDRESS?
S20 8044 S WITHOUT OR LESS OR "NOT" OR MISSING? OR NO OR UN OR NON OR LACKING OR ABSENT OR FREE OR MINUS OR ELIMINAT? OR SKIP? OR OMIT?
S21 30 S COULDN? OR WOULDN? OR SHOULDN? OR DOESN? OR CAN()T OR WON()T
S22 24 S UNMODULAT? OR NONMODULAT? OR UNCONVER? OR NONCONVER? OR UNMODIF? OR NONMODIF? OR NONALTER? OR UNALTER? OR NONTRANSFORM? OR UNTRANSFORM?
S23 91 S UNCHANG? OR NONCHANG? OR IMMUTAB? OR NONTRANSLAT? OR UNTRANSLAT? OR UNAMEND? OR NONAMEND? OR UNEDIT? OR NONEDIT? OR NONUPDAT? OR UNUPDAT?
S24 0 S UNMANIPULAT? OR NONMANIPULAT? OR UNCUSTOM? OR NONCUSTOM? OR UNPERSONALI? OR NONPERSONALI? OR NONINDIVIDUALI? OR UNINDIVIDUALI?
S25 2 S UNADAPT? OR NONADAPT? OR UNRECONFIGUR? OR NONRECONFIG? OR NONCONFIGUR? OR UNCONFIGUR? OR UNREVIS? OR NONREVIS? OR NONREARRANG? OR UNREARRANG?
S26 0 S MODULATIONLESS? OR MODULATIONFREE? OR CHANGEFREE? OR CHANGELESS? OR MODIFICATIONLESS? OR MODIFICATIONFREE?
S27 0 S AMENDMENTLESS? OR AMENDMENTFREE? OR EDITFREE? OR EDITLESS? OR UPDATELESS? OR UPDATEFREE? OR REVISIONLESS? OR REVISIONFREE?
S28 1595 S SOURCE? OR ISSUER? OR ISSUING? OR SEMINAL?
S29 2461 S NUMBER()(ONE OR 1) OR PRINCIPAL? OR LEAD OR CONTROLLER? OR HEAD OR MASTER OR BASIC?
S30 5217 S FIRST? OR 1ST OR PRIMARY OR INITIAL? OR ORIGINAL? OR LEADOFF? OR MAIN OR CHIEF OR INTRODUCTORY?
S31 2591 S TARGET? OR RECEIV? OR RECEPT? OR RECIPIENT? OR DESTINATION? OR ENDPOINT?
S32 2845 S SECOND? OR 2ND OR ANOTHER OR AUXILIAR? OR BACKUP? OR EXTRA OR SLAVE? OR SUPPLEMENT?
S33 4810 S SUBSIDIAR? OR DIFFERENT? OR ALTERNAT? OR NUMBER()(TWO OR 2)
S34 32 S S13 AND S6:S12(5N)S14:S19 AND S1(5N)(S20:S21(5N)S14:S19 OR S22:S27)

```
S35      25  S S34 AND S28:S33
S36      32  S S34:S35
S37      17  S S36 AND PY=1970:2002
S38      17  S S36 NOT PY=2003:2006
S39      17  S S37:S38
S40      11  RD  (unique items)
; show files
```

[File 2] **INSPEC** 1898-2006/Jun W1

[File 6] **NTIS** 1964-2006/Jun W1

[File 8] **Ei Compendex(R)** 1970-2006/Jun W1

[File 34] **SciSearch(R) Cited Ref Sci** 1990-2006/Jun W1

[File 35] **Dissertation Abs Online** 1861-2006/May

[File 56] **Computer and Information Systems Abstracts** 1966-2006/Jun

[File 60] **ANTE: Abstracts in New Tech & Engineer** 1966-2006/Jun

[File 62] **SPIN(R)** 1975-2006/Mar W4

[File 65] **Inside Conferences** 1993-2006/Jun 12

[File 94] **JICST-EPlus** 1985-2006/Mar W2

[File 95] **TEME-Technology & Management** 1989-2006/Jun W2

[File 99] **Wilson Appl. Sci & Tech Abs** 1983-2006/Apr

[File 111] **TGG Natl.Newspaper Index(SM)** 1979-2006/Jun 02

[File 144] **Pascal** 1973-2006/May W3

[File 239] **Mathsci** 1940-2006/Jul

Set Items Description
S1 1616404 S MAP? ? OR MAPP???
S2 4509371 S GUI? ? OR (GRAPHIC? OR VISUAL?)(2W)(INTERFACE? OR APP? ? OR APPLICAT?) OR DISPLAY? OR MENU? ? OR WEBPAGE?
S3 196119 S COMPUTER?()(SCREEN? OR WINDOW?) OR COMPUTER?(2N)(IMPLEMENT? OR TOOL? ? OR UTILIT?)
S4 120221 S USERINTERFAC? OR (DROP OR PULL)()DOWN? OR (WINDOW OR DESKTOP)()MANAGER? OR PULLDOWN?
S5 345824 S UI OR UIS OR (GRAPHIC? OR VISUAL? OR INFORMATION? OR DATA?)(2N)(MENU? OR PAGE? OR SCREEN?)
S6 9396475 S SOFTWARE()OBJECT? OR DATABASE? OR DOMAIN? OR COLUMN? OR TABLE? OR DATAOBJECT? OR DATASTRUCTUR? OR DATAVALUE?
S7 4704268 S OBJECT? OR DATAOBJECT? OR DATA()(ENTIT? OR STRUCTUR?) OR OBJECTMODEL? OR OBJECT()MODEL?
S8 26233 S DATACOMMAND? OR DATAORDER? OR DATAUNIT? OR DATA()UNIT? OR DATA()MESSAG? OR DATA()INSTRUCT?
S9 83583 S DATABYTE? OR DATABLOCK? OR DATA()(BLOCK? OR REQUEST? OR INPUT? OR TYPE? OR MEMBER? OR VALUE?) OR DATATYPE?
S10 1179410 S (DATA? OR SOFTWARE? OR OBJECT?)(2N)(IDENTIFIER? OR TAG? ? OR METADATA? OR ADDRESS? OR LOCATER? OR LABEL? OR NAME? OR EXTENSION? OR DESIGNATION?)
S11 1504362 S DATA()(REPRESENTATION? OR DEPOSITOR? OR REPOSITOR? OR BANK? OR FILE? OR COLLECTION?) OR ROW OR ROWS
S12 24214505 S LOG OR LOGS OR CATALOG? OR CHRONICL? OR RECORD? ? OR LIST??? OR FIELD? OR ARRAY? OR ARCHIV?
S13 18407 S S1(10N)S2:S5 AND S1:S5(10N)S6:S12
S14 8186 S MODULAT? OR CONVERT? OR MODIF? OR ALTER? OR TRANSFORM?
S15 13106 S CHANGE? OR CHANGING? OR TRANSLAT? OR AMEND? OR EDIT? OR UPDAT?
S16 6914 S CONVERSION? OR MANIPULAT? OR CUSTOMIZ? OR PERSONALIZ? OR INDIVIDUALIZ?
S17 6789 S CUSTOMIS? OR ADAPT? OR RECONFIGUR? OR CONFIGUR? OR REVIS? OR REARRANG?
S18 182 S PERSONALIS? OR (CUSTOM OR TAILOR)()(MAKE? OR MADE? OR MAKING)
S19 799 S REORDER? OR RESHUFFL? OR RESET? OR REFORMAT? OR READDRESS?
S20 15471 S WITHOUT OR LESS OR "NOT" OR MISSING? OR NO OR UN OR NON OR LACKING OR ABSENT OR FREE OR MINUS OR ELIMINAT? OR SKIP? OR OMIT?
S21 5168 S COULDN? OR WOULDN? OR SHOULDN? OR DOESN? OR CAN()T OR WON()T
S22 121 S UNMODULAT? OR NONMODULAT? OR UNCONVER? OR NONCONVER? OR UNMODIF? OR NONMODIF? OR NONALTER? OR UNALTER? OR NONTRANSFORM? OR UNTRANSFORM?
S23 304 S UNCHANG? OR NONCHANG? OR IMMUTAB? OR NONTRANSLAT? OR UNTRANSLAT? OR UNAMEND? OR NONAMEND? OR UNEDIT? OR NONEDIT? OR NONUPDAT? OR UNUPDAT?
S24 13 S UNMANIPULAT? OR NONMANIPULAT? OR UNCUSTOM? OR NONCUSTOM? OR UNPERSONALI? OR NONPERSONALI? OR NONINDIVIDUALI? OR UNINDIVIDUALI?
S25 8 S UNADAPT? OR NONADAPT? OR UNRECONFIGUR? OR NONRECONFIG? OR NONCONFIGUR? OR UNCONFIGUR? OR UNREVIS? OR NONREVIS? OR NONREARRANG? OR UNREARRANG?
S26 1 S MODULATIONLESS? OR MODULATIONFREE? OR CHANGEFREE? OR CHANGELESS? OR MODIFICATIONLESS? OR MODIFICATIONFREE?
S27 0 S AMENDMENTLESS? OR AMENDMENTFREE? OR EDITFREE? OR EDITLESS? OR UPDATELESS? OR UPDATEFREE? OR REVISIONLESS? OR REVISIONFREE?
S28 6670 S SOURCE? OR ISSUER? OR ISSUING? OR SEMINAL?
S29 8906 S NUMBER()(ONE OR 1) OR PRINCIPAL? OR LEAD OR CONTROLLER? OR HEAD OR MASTER OR BASIC?
S30 13287 S FIRST? OR 1ST OR PRIMARY OR INITIAL? OR ORIGINAL? OR LEADOFF? OR MAIN OR CHIEF OR INTRODUCTORY?
S31 9154 S TARGET? OR RECEIV? OR RECEPT? OR RECIPIENT? OR DESTINATION? OR ENDPOINT?
S32 10801 S SECOND? OR 2ND OR ANOTHER OR AUXILIAR? OR BACKUP? OR EXTRA OR SLAVE? OR SUPPLEMENT?
S33 9847 S SUBSIDIAR? OR DIFFERENT? OR ALTERNAT? OR NUMBER()(TWO OR 2)
S34 247 S S13 AND S14:S19(5N)S6:S12 AND (S20:S21(5N)S14:S19 OR S22:S27)(5N)S1

S35 213 S S34 AND (S28:S30 AND S31:S33)(5N)S1:S12
S36 247 S S34:S35
S37 213 S S36 AND PY=1970:2002
S38 213 S S36 NOT PY=2003:2006
S39 213 S S37:S38
S40 149 RD (unique items)
; show files

[File 9] **Business & Industry(R)** Jul/1994-2006/Jun 09
(c) 2006 The Gale Group. All rights reserved.

[File 13] **BAMP** 2006/Jun W1
(c) 2006 The Gale Group. All rights reserved.

[File 15] **ABI/Inform(R)** 1971-2006/Jun 12
(c) 2006 ProQuest Info&Learning. All rights reserved.

[File 16] **Gale Group PROMT(R)** 1990-2006/Jun 12
(c) 2006 The Gale Group. All rights reserved.

[File 20] **Dialog Global Reporter** 1997-2006/Jun 13
(c) 2006 Dialog. All rights reserved.

[File 47] **Gale Group Magazine DB(TM)** 1959-2006/Jun 12
(c) 2006 The Gale group. All rights reserved.

[File 75] **TGG Management Contents(R)** 86-2006/Jun W1
(c) 2006 The Gale Group. All rights reserved.

[File 88] **Gale Group Business A.R.T.S.** 1976-2006/Jun 05
(c) 2006 The Gale Group. All rights reserved.

[File 98] **General Sci Abs** 1984-2005/Jan
(c) 2006 The HW Wilson Co. All rights reserved.

[File 141] **Readers Guide** 1983-2006/Feb
(c) 2006 The HW Wilson Co. All rights reserved.

[File 148] **Gale Group Trade & Industry DB** 1976-2006/Jun 12
(c)2006 The Gale Group. All rights reserved.

[File 160] **Gale Group PROMT(R)** 1972-1989
(c) 1999 The Gale Group. All rights reserved.

[File 275] **Gale Group Computer DB(TM)** 1983-2006/Jun 12
(c) 2006 The Gale Group. All rights reserved.

[File 369] **New Scientist** 1994-2006/Jun W1
(c) 2006 Reed Business Information Ltd. All rights reserved.

[File 370] **Science** 1996-1999/Jul W3

File 370: This file is closed (no updates). Use File 47 for more current information.

[File 484] **Periodical Abs Plustext** 1986-2006/Jun W1

[File 553] **Wilson Bus. Abs.** 1982-2006/Jun

[File 610] **Business Wire** 1999-2006/Jun 13

File 610: File 610 now contains data from 3/99 forward. Archive data (1986-2/99) is available in File 810.

[File 613] **PR Newswire** 1999-2006/Jun 13

File 613: File 613 now contains data from 5/99 forward. Archive data (1987-4/99) is available in File 813.

[File 621] **Gale Group New Prod.Annou.(R)** 1985-2006/Jun 12

[File 624] **McGraw-Hill Publications** 1985-2006/Jun 13

File 624: Homeland Security & Defense and 9 Platt energy journals added Please see HELP NEWS624 for more

[File 634] **San Jose Mercury** Jun 1985-2006/Jun 12

[File 635] **Business Dateline(R)** 1985-2006/Jun 10

[File 636] **Gale Group Newsletter DB(TM)** 1987-2006/Jun 12

[File 647] **CMP Computer Fulltext** 1988-2006/Jul W3

[File 674] **Computer News Fulltext** 1989-2006/Jun W2

[File 696] **DIALOG Telecom. Newsletters** 1995-2006/Jun 12

[File 810] **Business Wire** 1986-1999/Feb 28

[File 813] **PR Newswire** 1987-1999/Apr 30

40/3,K/119 (Item 3 from file: 275) **Links**
Gale Group Computer DB(TM)

01937710 **Supplier Number:** 18270752 **(Use Format 7 Or 9 For** FULL TEXT)
Synlib: the core of CDE tests.(Common Desktop Environment) (Technology Information)(Technical)

Chakrabarti, Sankar L.
Hewlett-Packard Journal , v47 , n2 , p62(4)
April , 1996
Document Type: Technical
ISSN: 0018-1153
Language: English **Record Type:** Fulltext; Abstract
Word Count: 2735 **Line Count:** 00231

Abstract: ...to test every property of a target application. Synlib reduces the arduous task of testing **GUI** software to creating individual test programs. The API uses focus **maps** and focus paths to **name GUI objects** of interest. Synlib is completely platform independent. This is an important consideration because the tests...

...Synlib API. These programs in turn manipulate and verify the appearance and behavior of the **GUI** application of interest (the **target** application). The recommended approach is to create many small Synlib programs to test the **target** software:

1. View the **target GUI** software as a collection of implemented properties. 2. Create a Synlib-based program to verify each property. 3. A collection of such Synlib programs forms a test suite for the **GUI** software.

With Synlib, the **main** task of testing **GUI** software reduces to creating the individual test programs. Considering each test program to be your...

...Synlib supports both alternatives. 3. Make sure that a new style manager window is now **displayed**. If the style manager window is **mapped** then the agent should report that the test passed. Otherwise, the agent should report that...

```
...icon"); /*
     * In the alternate implementation using focus
     * maps we could simply say:
     * result = SynSelectItem (dpy,
     * "MyFrontPanel",
     * "StyleManager...printf ("Test Aborted: Front Panel window
     could not be found. \n"); }
```

SynOpenDisplay() connects to the **display** on which the **target** application is running or will run. SynNameWindowByTitle() determines if a window of the specified title...

...icon. The agent is being asked to expect and wait for a window to **map** on the **display**. The function SynWaitWindowMap accomplishes the wait and names the window StyleManager if and when a window **maps** on the **display**. If the **mapped** window has

the expected title, StyleManager, then the agent is to conclude that the test...

...Nonetheless, this simple program illustrates the basic paradigm for creating test agents. You need to **name** the **GUI objects** of interest and provide a mechanism for the agent to identify these objects during execution. The agent should be able to **manipulate** the **named objects** by delivering keystrokes or button inputs. Finally, the agent should be able to verify if...

...to provide all of these capabilities. Table I is a summary of Synlib's capabilities.

Table 1
Synlib Capabilities

Functions to **Name GUI Objects** of Interest
- SynNameWindow
- SynNameWindowByTitle
- SynNameLocation
- SynNameRegion

Functions to Deliver Inputs to Named Objects
- SynClickBunon
- SynClickKey...

...above there is nothing that is tied to specific features of a platform or a **display** on which the **target GUI** application may be executing. All platform dependent information can be (and is encouraged to be...

...window - is not present in the program but is declared in a file called the **object map**. At execution time the **object map** is made available to the Synlib agent through a command line option. The agent consults the **object map** to decode the exact location of the **named object** styleManager...

...unchanged even if the front panel configuration changes or if the exact location of the **named object** is **different** on a **different display**. The named location, styleManager...
...icon, represents a semantic entity whose meaning is independent of the platform or the **display** or the revision of the **target** application. The semantics of the name is meaningful only in the test. In other words, the test remains portable. If changes in the platform, **display**, or application require that the exact location of the **named object** be **changed**, this is achieved either by **editing** the **object map** file or by supplying a **different object map** file specific for the platform. Synlib provides automated methods to **edit** or generate environment-specific **object map** files. The agent itself does **not** need any **change**. The format of a typical Synlib **object map** is:

object Map for the sample program Declares the locations named in the test program Location styleManagerFocusPath
FrontPanel.ActionIcons.dtstyleIcon

Focus **Maps**

A far superior method of naming **GUI objects** of interest is to use Synlib's concepts of focus **maps** and focus paths. A focus map is a description of the logical organization of the...

...separated names declared in the focus map named FrontPanel Focus maps are described in focus **map** files. Focus paths, on the other hand, are declared in **object map** files because, when associated with a window, a focus path identifies an actual object capable...
...FocusPath, which can be declared in the **object map** file as shown above.

The following is the focus **map** for the front panel window.

```
! Focus map for the default front panel window of the...

...Panel helpIcon trashIcon
    dtcmIcon dtfileIcon))) !
```

If the proper focus **map** and **object maps** are provided, the agent will apply Synlib embedded rules to decide how to set focus on the named item and then select or activate the item. During execution, Synlib **first** processes all supplied focus **maps** and creates an internal representation. Whenever the program refers to a focus path, Synlib decodes the identity of the desired **object** by analyzing the focus **map** in which the focus path occurs. Using the declarations in the focus map and applying...a simple situation, the same principles and methods can with equal ease be used to **name** and access **objects** in deeply embedded structures like **menus** and submenus. In general, naming **objects** by means of a focus **map** is far superior to naming them by means of an **object map**. Because access to the **objects** of interest is via a dynamically generated sequence of keystrokes, the programs employing these methods are resistant to changes in window size, fonts, or actual **object** locations. This makes the tests completely portable across platforms, **displays**, and other environmental variabilities. Synlib programs using focus **maps** to **name GUI objects** need **not** be **changed** at all unless the specification of the target application changes.

Using a similar soft coding...

...test for one application." The tests, assisted by the required environment dependent resource files like **object map**, focus **map**, and key **map** files, can verify the behavior of target applications executing on **different** platforms, using **different displays**, and working in very **different** language environments.

Fig. 2 shows an execution architecture for Synlib tests. A key map file...

Synlib: The Core of CDE Tests

Synlib is an application program interface for creating tests for graphical user interface applications. A collection of Synlib programs, each designed to verify a specific property of the target software, forms a test suite for the application. Synlib tests can be completely platform independent—an advantage for testing the Common Desktop Environment (CDE), which runs on the platforms of the four participating companies.

by Sankar L. Chakrabarti

Synlib is a C-language application program interface (API) designed to enable the user to create tests for graphical user interface (GUI) software. The user simply tells Synlib what to do and it will execute the user-specified tests on a variety of displays and in a variety of execution environments.

A complete description of the Synlib API is beyond the scope of this article and can be found in references 1 through 4. In this article we will only indicate how the ideas and capabilities supported by Synlib were applied to the development of tests for the Common Desktop Environment (CDE) described in *Article 1*. An overview of CDE test technology including a description of the role played by Synlib can be found in *Article 7*.

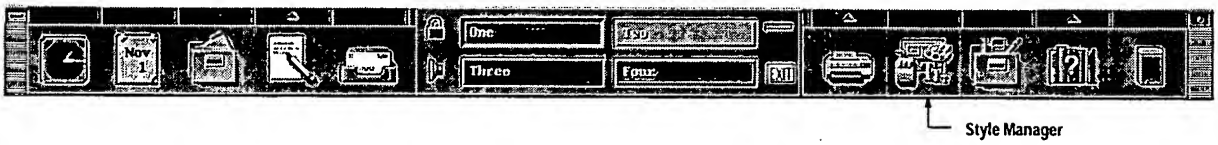
To test GUI software, we create programs using the Synlib API. These programs in turn manipulate and verify the appearance and behavior of the GUI application of interest (the target application). The recommended approach is to create many small Synlib programs to test the target software:

1. View the target GUI software as a collection of implemented properties.
2. Create a Synlib-based program to verify each property.
3. A collection of such Synlib programs forms a test suite for the GUI software.

With Synlib, the main task of testing GUI software reduces to creating the individual test programs. Considering each test program to be your agent, your task is to tell the agent what to do to verify a specific property of the program you wish to test. Assume that you want to test the following property of the front panel on the CDE display (Fig. 1): *On clicking the left mouse button on the style manager icon on the CDE front panel, the style manager application will be launched.* It is likely that you will tell your agent the following to verify if the property is valid for a given implementation of the front panel:

1. Make sure that the front panel window is displayed on the display.
2. Click the style manager icon on the front panel window. Alternatively, you might ask the agent to select the style manager icon on the front panel and leave it to the agent to decide how to select the icon. Synlib supports both alternatives.
3. Make sure that a new style manager window is now displayed. If the style manager window is mapped then the agent should report that the test passed. Otherwise, the agent should report that the test failed.

Fig. 1. Front panel of the CDE desktop.



These instructions are captured in the following Synlib program. All function calls with the prefix Syn are part of the Synlib API.

```
main(argc, argv)
int argc;
char **argv;
{
 Display *dpy;
 int windowCount;
 Window *windowList;
 Window shell_window;
 char *title;
 SynStatus result;

 dpy = SynOpenDisplay(NULL);
 result = SynNameWindowByTitle (dpy,
  "MyFrontPanel", "One", PARTIAL_MATCH,
  &windowCount, &windowList);
if (result == SYN_SUCCESS)
 {
  result = SynClickButton(dpy, "Button1",
   "MyFrontPanel", "styleManager_icon");
  /*
   * In the alternate implementation using focus      *  maps we could simply say:
   * result = SynSelectItem (dpy,
   *  "MyFrontPanel",
   *  "StyleManager_Icon_FocusPath");
   */
  result = SynWaitWindowMap (dpy,
    "StyleManager",TIME_OUT);
  if (result == SYN_SUCCESS)
   { result = SynGetShellAndTitle (dpy,
     "Style Manager", &shell_window, &title);
      if (strcmp (title, "Style Manager") ==          0)
       printf ("Test Passed: Style Manager
       window appeared. \n");
    else
       printf ("Test Failed: Expected Style
        Manager window; found %s\n",title);
   }
   else
    printf ("Test Failed: Expected Style Manager
      window would map but none did.\n");
 }
 else
  printf ("Test Aborted: Front Panel window         could not be found. \n");
}
```

SynOpenDisplay() connects to the display on which the target application is running or will run. SynNameWindowByTitle() determines if a window of the specified title (in this case One) is already displayed. If so, the user (i.e., the programmer) chooses to name it MyFrontPanel. Through SynClickButton(), the Synlib agent is instructed to click on the window MyFrontPanel at a location called styleManager_icon. The agent is being asked to expect and wait for a window to map on the display. The function SynWaitWindowMap accomplishes the wait and names the window StyleManager if and when a window maps on the display. If the mapped window has the expected title, StyleManager, then the agent is to conclude that the test succeeded, that is, the front panel window had the specified property (clicking on the style manager icon really launched the style manager application).

In practice, the tests would be more complicated than this example. Nonetheless, this simple program illustrates the basic paradigm for creating test agents. You need to name the GUI objects of interest and provide a mechanism for the agent to identify these objects during execution. The agent should be able to manipulate the named objects by delivering keystrokes or button inputs. Finally, the agent should be able to verify if specified things happened as a result of processing the delivered inputs. Synlib is designed to provide all of these capabilities. Table I is a summary of Synlib's capabilities.

Platform Independence

Synlib programs can be written so that they are completely platform independent. For example, in the program above there is nothing that is tied to specific features of a platform or a display on which the target GUI application may be executing. All platform dependent information can be (and is encouraged to be) abstracted away through the mechanism of *soft coding*. In the program above, the statement using the function SynClickButton is an example of soft coding. The last parameter in this statement, styleManager_icon, refers to a location in the front panel window. The exact definition of the location—the window's x,y location with respect to the FrontPanel window—is not present in the program but is declared in a file called the *object map*. At execution time the object map is made available to the Synlib agent through a command line option. The agent consults the object map to decode the exact location of the named object styleManager_icon, then drives the mouse to the decoded location and presses the button. Because the location is soft coded, the program itself remains unchanged even if the front panel configuration changes or if the exact location of the named object is different on a different display. The named location, styleManager_icon, represents a semantic entity whose meaning is independent of the platform or the display or the revision of the target application. The semantics of the name is meaningful only in the test. In other words, the test remains portable. If changes in the platform, display, or application require that the exact location of the named object be changed, this is achieved either by editing the object map file or by supplying a different object map file specific for the platform. Synlib provides automated methods to edit or generate environment-specific object map files. The agent itself does not need any change.

Table 1
Synlib Capabilities

Functions to Name GUI Objects of Interest
- SynNameWindow
- SynNameWindowByTitle
- SynNameLocation
- SynNameRegion

Functions to Deliver Inputs to Named Objects
- SynClickButton
- SynClickKey
- SynPressAndHoldButton
- SynReleaseButton
- SynMovePointer
- SynPrintString
- SynPressAndHoldKey
- SynReleaseKey
- SynSetFocus
- SynSelectItem

Functions to Synchronize Application State with Test Agent
- SynWaitWindowMap
- SynWaitWindowUnmap
- SynWaitWindowConfigure
- SynWaitProperty

Functions to Verify the State of a GUI Application
- SynGetShellAndTitle
- SynStoreText
- SynCompareWindowImage

Miscellaneous Functions to Process Needed Test Resources from the Environment
- SynParseCommandOptions
- SynParseObjectFile
- SynBuildFocusMap
- SynParseKeyMap

The format of a typical Synlib object map is:

```
! Object Map for the sample program
! Declares the locations named in the test
!   program
Location styleManager_icon 781 51

! Declares the full path of an item named in a    !    focus map
FocusPath StyleManager_Icon_FocusPath
  FrontPanel.ActionIcons.dtstyleIcon
```

Focus Maps

A far superior method of naming GUI objects of interest is to use Synlib's concepts of *focus maps* and *focus paths*. A focus map is a description of the logical organization of the input enabled objects in a widget-based application. An input enabled object is a region in a window that can accept keystrokes or button inputs from a user. Generally these objects are widgets or gadgets used in constructing the user interface.

The method of constructing a focus map is fully described in the Synlib User's Guide.[1] A more complete description of the concept of a focus map and its use in testing X windows applications has been published elsewhere.[2] A focus path is a string of dot-separated names declared in a focus map. For example, StyleManager_Icon_FocusPath is the name of the focus path FrontPanel.ActionIcons.dtstyleIcon, which is a string of dot-separated names declared in the focus map named FrontPanel. Focus maps are described in focus map files. Focus paths, on the other hand, are declared in object map files because, when associated with a window, a focus path identfies an actual object capable of accepting input.

In the example program above, the function SynSelectItem() represents an instruction to the agent to select the object named by the string StyleManager_Icon_FocusPath, which can be declared in the object map file as shown above.

The following is the focus map for the front panel window.

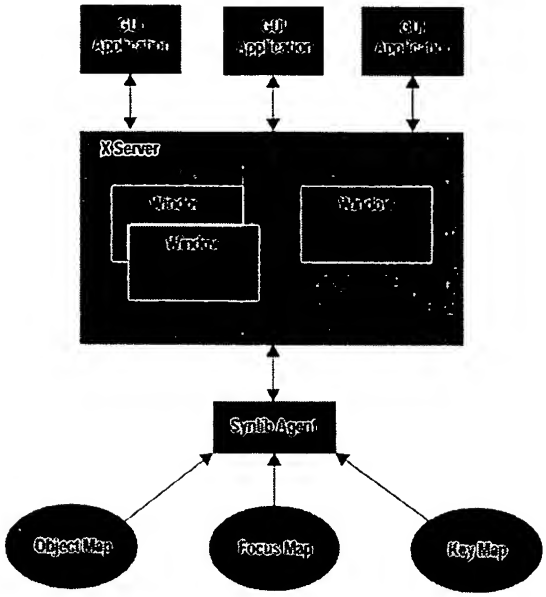
```
!
! Focus map for the default front panel window
!   of the CDE desktop.
!
(FocusMap FrontPanel
 (FocusGroup ActionIcons
    (App_Panel dtpadIcon dtmailIcon dtlockIcon
    dtbeepIcon workspace_One workspace_Three
    workspace_Two workspace_Four exitIcon
    printer_Panel printerIcon dtstyleIcon
    toolboxIcon Help_Panel helpIcon trashIcon
    dtcmIcon dtfileIcon)))
!
```

If the proper focus map and object maps are provided, the agent will apply Synlib embedded rules to decide how to set focus on the named item and then select or activate the item. During execution, Synlib first processes all supplied focus maps and creates an internal representation. Whenever the program refers to a focus path, Synlib decodes the identity of the desired object by analyzing the focus map in which the focus path occurs. Using the declarations in the focus map and applying OSF/Motif supported keyboard traversal specifications, Synlib generates a series of keystrokes to set the keyboard focus to the object indirectly named via the focus path. The rules for transforming the focus path name to the sequence of keystrokes are somewhat complex and have been fully described elsewhere.[2] These rules are embedded in Synlib and are completely transparent to the user.

This example shows the use of a focus map in naming icons in the front panel. Although the example here deals with a simple situation, the same principles and methods can with equal ease be used to name and access objects in deeply embedded structures like menus and submenus. In general, naming objects by means of a focus map is far superior to naming them by means of an object map. Because access to the objects of interest is via a dynamically generated sequence of keystrokes, the programs employing these methods are resistant to changes in window size, fonts, or actual object locations. This makes the tests completely portable across platforms, displays, and other environmental variabilities. Synlib programs using focus maps to name GUI objects need not be changed at all unless the specification of the target application changes.

Using a similar soft coding technique, Synlib makes it possible to create *locale neutral* tests, that is, tests that can verify the behavior of target applications executing in different language environments without undergoing any change themselves. Use of this technique has substantially reduced the cost of testing internationalized GUI applications. A complete description of the concept of locale neutral tests has been published.[4]

Fig. 2. Synlib test execution architecture.



Test Execution Architecture

Synlib provides concepts and tools that enable us to create "one test for one application." The tests, assisted by the required environment dependent resource files like object map, focus map, and key map files, can verify the behavior of target applications executing on different platforms, using different displays, and working in very different language environments.

Fig. 2 shows an execution architecture for Synlib tests. A key map file contains declarations to name keystrokes, button events, and sequences of keystrokes and button events. The key map file provides a way to virtualize and name all inputs to be used by a test program. This mechanism is very useful for creating tests for internationalized applications and is fully described in reference 4.

The cost of creating or modifying the environment resource files is minuscule compared to the cost of creating the tests themselves. Thus, the ability to create tests that are insensitive to differences in the execution environment of the target application has been a great productivity boost to our testing efforts.

A feature of Synlib test technology is that it does not require any change in the target application. It does not require that the application code be modified in any way. There is no need to incorporate any test hook in the application, nor is the application required to relink to any foreign test-specific library. Synlib supports a completely noninvasive testing framework. The test is directly executed on the application off the shelf. Synlib even makes it possible to write the tests before the application is ready for testing.[3]

The author originally designed Synlib to solve the problems of GUI testing facing our lab, mainly testing GUI applications that supported a variety of HP displays and operating systems. We designed Synlib to provide a technology that yields robust and platform-insensitive tests at a low cost. Synlib proved to be a marvelous fit for testing the CDE desktop since one of the main conditions was that the tests would have to verify applications running on the platforms of the four participating companies. Essentially it was a problem of creating platform-insensitive tests, a problem that we had already solved. The success of Synlib in this endeavour is shown by the large body of functioning test suites for the complex applications of the CDE desktop.

Acknowledgments

The development of Synlib has benefited from the comments, criticism, and support of many people. The author wishes to thank everyone who willingly came forward to help mature this technology. Harry Phinney, Fred Taft, and Bill Yoder were the first to create test suites for their products using Synlib. Their work proved the value of Synlib to the rest of the laboratory. Subsequently, Bob Miller allowed his group to experiment with Synlib, which led to its adoption as the testing tool for CDE. Thanks Bob, Harry, Fred, and Bill. Julie Skeen and Art Barstow volunteered their time to review the initial design of Synlib. A very special thanks is due Julie. In many ways the pleasant and intuitive user interface of Synlib can be traced to her suggestions. Thanks are also due the engineers at the multimedia lab who proved the effectiveness of Synlib in testing multimedia applications. Ione Crandell empowered this effort. Kristann Orton wrote many of the man pages. Dennis Harms and Paul Ritter effectively supported Synlib in many stormy CDE

sessions. Thanks Dennis, Kritann, and Paul. Michael Wilson taught me how Synlib could solve the knotty problem of testing hyperlinked systems. Thanks, Mike. Claudia DeBlau and Kimberly Baker, both of Sun Microsystems, helped in developing Synlib's interface to the Test Environment Toolkit (TET). Finally the author thanks Ken Bronstein. Ken appreciated the value of this unofficial job from the very beginning. Ken's unwavering support has been crucial to the continued development of this technology.

References

1. S.L. Chakrabarti, *Synlib User's Guide—An Experiment in Creating GUI Test Programs*, Hewlett-Packard Company.
2. S.L. Chakrabarti, "Testing X Clients Using Synlib and Focus Maps," *The X Resource*, Issue 13, Winter 1995.
3. S.L. Chakrabarti, R. Pandey, and S. Mohammed, "Writing GUI Specifications in C," *Proceedings of the International Software Quality Conference*, 1995.
4. S.L. Chakrabarti and S. Girdhar, "Testing 'Internationalized' GUI Applications," accepted for publication, *The X Resource*, Winter 1996.

▶ Go to Article 9
▶ Go to Table of Contents
▶ Go to HP Journal Home Page

40/3,K/117 (Item 1 from file: 275) **Links**

02190717 **Supplier Number:** 20584567 **(Use Format 7 Or 9 For** FULL TEXT)

Linking enterprise business systems to the factory floor. (information driving business enterprises) (Product Information)(Technical)

Jennyc, Kenn S.
Hewlett-Packard Journal , v49 , n2 , p62(12)
May , 1998
Document Type: Technical
ISSN: 0018-1153
Language: English **Record Type:** Fulltext; Abstract
Word Count: 5494 **Line Count:** 00465

Abstract: ...direct data movement between two software applications, a background data server and a set of **configuration** files. It contains communication **objects** and a **GUI** for **mapping** program functions among business and manufacturing programs. The program also performs spooling and tracing and...

...that have proprietary hardware architectures and operating systems is often not usable on other systems.

* **Different** applications use **different data types** according to their specific needs.

* Incompatible **data structures** often result because of the **different** groupings of data elements by software applications. For example, an element with a common logical...

...in two applications may still be stored with two different physical representations.

* Applications written in **different** languages sometimes interpret their **data values differently**. For example C and COBOL interpret binary numeric **data values differently**.

What is needed, therefore, is an off-the-shelf product that is specifically designed to...

...Link components listed above have the common goal of enabling users to create middleware that **maps** components with **different** interfaces together for data transfer.

In HP Enterprise Link, the combination of a single source address and a single **destination** address is called a **mapping**. A unit of data at the specified **source** address is said to be **mapped** to the specified **destination** address. In other words, it can be read from the specified source address and written to the specified **destination** address.

Although a **mapping** deals with the transfer of a single unit of data, real-world situations usually require...

...circumstances such transfers should occur (Figure 3).

(Figure 3 ILLUSTRATION OMITTED)

The HP Enterprise Link **configuration** tool is composed of communication **objects** and a **graphical** user **interface** (**GUI**). Communication **objects** are used to obtain **namespace data** that is unique to each application and to provide application-specific windows. The configuration tool provides the user with an easy-to-use point-and-click style **GUI**.

All dependencies on particular software applications are encapsulated in communication **objects**. The **configuration** tool's communication **objects** provide the following functionality:

* They fetch namespace information from communicating software applications for presentation to...to specify triggers unique to a particular software application.

* They provide routines to tell the **GUI** exactly what functionality is supported by a communication **object**. For example, can the application software serve only as a data **source** (supply **data values**), or can it serve as both a data source and a data **destination** (both supply and use **data** values)?

There are three important windows in the configuration tool's **GUI**: the Edit Method window, the Edit **Mapping** window, and the Trigger Configuration window.

Edit Mapping. The Edit Mapping window is used to...

...users to specify which data to move where. They don't have to remember the **names** of **data sources** or data **destinations**. Instead they just choose from the **displayed list** of possibilities. The side-by-side **display** of application namespaces makes it much easier to integrate the applications.

(Figure 4 ILLUSTRATION OMITTED...

...many software applications.

To create a new mapping the user selects an item from the **Mapping Source** tree diagram and an item from the **Mapping Destination** tree diagram, and then clicks the Add **Mapping** button. A new **mapping** is added to the **mapping table displayed** on the **Edit** Method window (Figure 5).

(Figure 5 ILLUSTRATION OMITTED)

Multiple static mappings can be created in...

...transform these addresses to valid destination addresses.

Edit Method. The Edit Method window (Figure 5) **displays** a method's mappings as a two-**column table** titled Mappings. **Source** addresses appear in the left **column** and **destination** addresses appear in the right. The data server transfers **mapped data** from **source addresses** to **destination** addresses in the same order as the mappings are listed in this table. The Mappings...

...in the source application. For the RTAP triggers in Figure 6 interesting events include a **database** value **change** or the occurrence of an RTAP **database** alarm. Data can also be **mapped** when something interesting happens in the destination application.

Thus, triggers allow data transfers to be...

...s methods.

The Data Server

The HP Enterprise Link data server is composed of communication **objects**, a trigger manager, and a **mapping** engine (Figure 7). Communication **objects** deal with the problems of generating triggers and getting data into and out of software...

...events. The mapping engine deals with the problems of reading configuration files, responding to triggers, **mapping source** addresses to **destination** addresses, and transforming the data as it is being mapped.

(Figure 7 ILLUSTRATION OMITTED)

All software-application dependencies are encapsulated in communication **objects**. Communication **objects** serve as **translators** between external software applications and the data server's mapping engine--they translate the software application's native application program interface (API) to the interface used by the **mapping** engine.

The interface between a communication **object** and the **mapping** engine is standardized, with all communication **objects** using the same interface. For data that is being transferred, the interface consists solely of...memory, shared files, TCP/IP sockets, or an application program interface (API).

When a communication **object** transfers data, it **translates** data between the format used by the source software application and the neutral format required by the **mapping** engine. For example, for numeric values, a communication **object** may have to **translate** between binary IEEE-754 floating-point format and the mapping engine's neutral format.

In...

...the current trigger event (if multiple methods were simultaneously triggered).

The interface between the communication **object** and the **mapping** engine is designed to support transaction-oriented data transfers, using commit and rollback. This functionality...

...waiting for the trigger criteria of any configured method to be satisfied.

When either a **source** or **destination** communication **object** in the data server detects that a method's trigger criteria have been satisfied, the...

...informs the data server trigger manager that a method has been triggered. This starts the **mapping** engine. **Alternatively**, if the data server trigger manager detects that a method's time-based trigger criteria have been satisfied, the mapping engine starts.

When triggered, the **mapping** engine requests that the **source** communication **object** provide the current **data values** at the method's **configured source** addresses. The **source** communication **object** obtains these values from the

software application, **translates** the format of all fetched **data values** to a neutral format, and passes the result to the **mapping** engine as address-value pairs, with one such pair for each of the method's defined mappings.

The data server **mapping** engine looks up the **destination** address that corresponds to each source address. This lookup results in a new list of address-value pairs, with the address now being the destination address, and the value **unchanged** (and still expressed in the **mapping** engine's neutral format). To minimize the impact on performance, this lookup is implemented using a hash **table**.

The **mapping** engine sends the new **list** of address-value pairs to the **destination** communication **object**. The **destination** communication **object converts** the **received** values into the format required by the destination software application, and writes the converted result to the specified addresses in the **destination** software application.

Communication **Objects** and Software Applications

There are two fundamental ways for software applications to provide communication objects...

...request-reply method, the communication object sends a software application the address of a wanted **data unit** in a request and **receives** its current value in a reply. With this method the communication object controls the data...

...of software applications that employ the request-reply method.

In the spontaneous-message method, communication **objects** **receive** data, usually as messages, from the software application whenever the application chooses to send it...

...received it. It also means that incoming data must either be safely transferred through the **mapping** engine or locally buffered when a communication **object** accepts data from the **source** application software.

Spooling is especially important if the source application is separated from the HP...forms in which trace results can be expressed include:

(Figure 8 ILLUSTRATION OMITTED)

* Data as **received** by a data server communication **object** from a **source** software application. This trace data is expressed using the source software application's native data...

...or read, and the time of transfer.

* Data as sent by a data server communication **object** to the **destination** software application. This trace data is expressed using the destination software application's native data...

...destination address, the value transferred, and the time of transfer.

Error messages reported by the **mapping** engine or by communication

objects can also be included in the trace output. This ability ensures that the relative sequencing...

...application can be interrupted. Interrupting the flow here allows the data server to read from **mapped source** addresses, **map** to new **destination** addresses, and then discard the data just before it would have been written to the...
...the flow here allows the data server to ignore all data sent to the communication **object** by the **source** software application.

Data Integrity

The HP Enterprise Link data server is carefully designed to preserve ...

...off.

Communication channel failures must be handled carefully. If the communication channel connecting a communication **object** to its software application fails, the data being **mapped** at the time of failure must not be lost or duplicated. Also, after normal operation...

...are taken to ensure data integrity when communication channels fail:

* For data received from the **source** software application, the communication **object** never acknowledges receipt of the data until the data has safely been saved to a disk-resident receive-spool file.
* Data **received** by the communication **object** from the **source** software application is not removed from the receive-spool file until the data has successfully passed through the **mapping** engine and been forwarded to the communication **object** responsible for sending it to the **destination** software application.
* The communication **object** that sends data to the **destination** software application only notifies the **mapping** engine that it successfully **received** the data after the data has been safely saved to a disk-resident transmitspool Me...

19980500

Linking Enterprise Business Systems to the Factory Floor

Kenn S. Jennyc

Information is the fuel that drives today's business enterprises. The ability to link different components in the enterprise together in a user-friendly and transparent manner increases the effectiveness of companies involved in manufacturing and production.

Computers have had a profound effect on how companies conduct business. They are used to run enterprise business software and to automate factory-floor production. While this has been a great benefit, the level of coordination between computers running unrelated application software is usually limited. This is because such data transfers are difficult to implement, often requiring manual intervention or customized software. Until recently, off-the-shelf data transfer solutions were not available.

HP Enterprise Link is a middleware software product that increases the effectiveness of companies involved in manufacturing and production. It allows business management software running at the enterprise level, such as SAP's R/3 product, to exchange information (via electronic transfer) with software applications running on the factory floor. It also allows software applications running on the factory floor to exchange information with each other.

HP Enterprise Link is available for HP 9000 computers running the HP-UX* operating system and PC platforms running Microsoft's Windows® NT operating system.

This article will discuss the evolution of the link between business software systems and factory automation systems, and the functionality provided in HP Enterprise Link to enable these two environments to communicate.

Kenn S. Jennyc
Kenn Jennyc is a software engineer at the HP Lake Stevens Division. He worked on the software design, development, and quality assurance for the HP Enterprise Link. Before that he worked on software design and development for the RTAP (real-time application platform) product. He received a BSEE degree from the University of Calgary in 1983 and came to HP in 1989. Kenn was born in Calgary, Alberta, Canada, is married, and has two children. In his spare time he likes to fly his home-built aircraft and dabble in analog electronics.

Background

Initially, only large corporations could afford computers. They ran batch-oriented enterprise business software to do payroll, scheduling, and inventory.

As the cost of computing dropped, smaller companies began using computers to run business software, and companies involved in manufacturing began using them to automate factory-floor production.

Although factory-floor automation led to improved efficiency and productivity, it was usually conducted on a piecemeal basis. Different portions of an assembly line were often automated at different times and often with different computer equipment, depending on the capabilities of computer equipment available at the time of purchase. As a result, today's factory-floor computers are usually isolated hosts, dedicated to automating selected steps in production. While various factory-floor functions are automated, they do not necessarily communicate with one another. They are isolated in "islands of automation." To make matters worse, the development of programmable logic controllers (PLCs) and other dedicated "smart" factory-floor devices has increased the number of isolated computers, making the goal of integrated factory-floor computation that much harder to achieve.

While production software was generally used for smaller, more isolated problems, business software was used to solve larger company-wide problems. Furthermore, while production software was more real-time oriented, business software was more transaction and batch oriented. These differing needs caused business systems to evolve with little concern for the kind of computing found on the factory floor. Similarly, production systems evolved with little concern for the kind of computing found at the enterprise level. As a result, many enterprise-level business systems and factory-floor computers are not able to intercommunicate. **Figure 1** shows an example of the components that make up a typical enterprise and factory-floor environment.

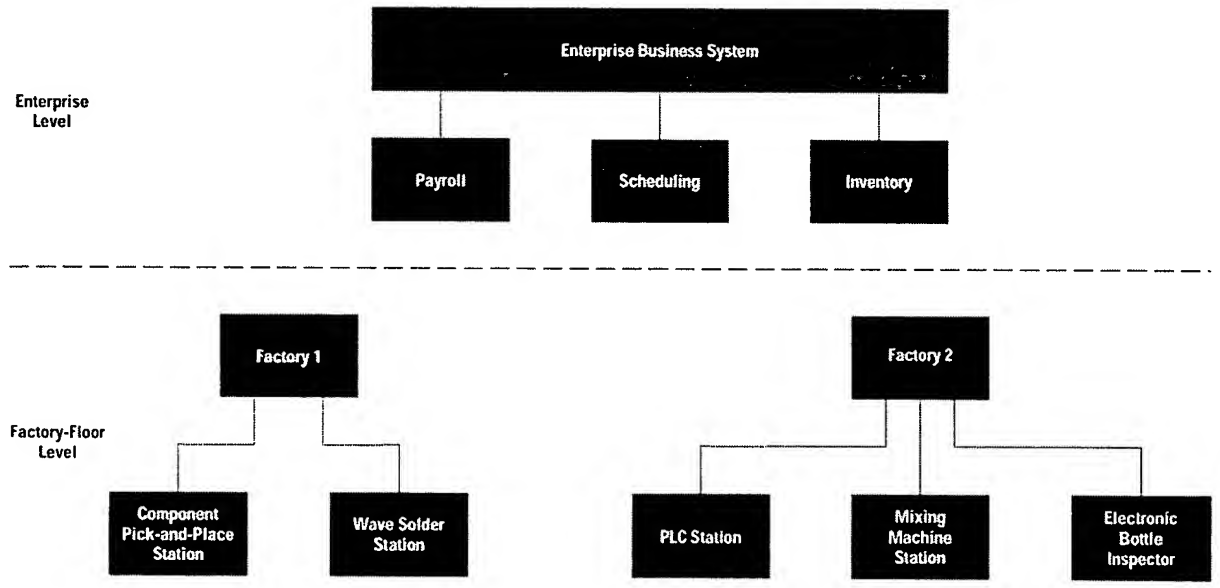
The net effect is that today companies find it difficult and expensive to integrate factory-floor systems with each other and with business software running at the enterprise level. This is unfortunate because the dynamic nature of the marketplace and the desire to reduce inventory levels have made the need for such integration very high.

Marketplace Dynamics

Over the last decade, the marketplace has become increasingly dynamic, forcing businesses to adapt ever more quickly to changing market conditions. Computer systems now experience a continuous stream of modifications and

Figure 1

Computing at the enterprise and factory-floor levels.



upgrades. Generally, this has forced business systems to adopt more real-time behaviors and production systems to become more flexible. It has also increased the frequency and volume of data transferred between business and production systems and between the many production systems.

There has always been a requirement to transfer information between computers in an organization, both horizontally between computers at the same functional level, and vertically between computers at different functional levels. In the past, manual data entry was an often-used approach. Hard-copy printouts generated by business management systems would be provided to operators who manually entered the information into one or more production systems. Although this was an acceptable approach in the past, such an approach is not sufficiently responsive in today's dynamic business environment. As a result, the need for electronic data transfer capability between the various business management and production level computers is now very high.

Electronic Data Transfers

Integrated business software with built-in support for data transfers between components is sometimes used at the business management level. While this minimizes the effort required to exchange data between the various components of enterprise business systems, it is often inflexible and restrictive with regard to what can be exchanged and when exchanges occur.

Organizations that use a variety of business software packages, rather than a single integrated package, have typically developed custom software for electronic data transfers between packages. Unfortunately, marketplace dynamics require custom software to be constantly reworked. This ongoing rework forces companies to either maintain in-house programming expertise or repeatedly hire software consultants to implement needed changes. As a result, custom data transfer software is not only expensive to develop but also costly to maintain—especially if changes must be implemented on short notice.

On the factory floor, software programmers have been employed to develop custom data transfer solutions that allow the different islands of automation to communicate. As previously noted, this approach is difficult to implement and expensive to maintain. In addition, this approach is often inflexible since the resulting software is usually developed assuming that the configuration of factory-floor systems is largely static.

When new equipment and application software are to be integrated into the overall system, software programmers don't just prepare additional custom software. They must also modify the existing custom software for all applications involved. For this reason, custom software is often avoided, and electronic data transfer capability is frequently confined to transfers between equipment and software supplied by the same manufacturer.

Differences in hardware (and associated operating systems) and differences in the software applications themselves cause numerous application integration problems. Here are a few examples:

- Data from applications running on computers that have proprietary hardware architectures and operating systems is often not usable on other systems.
- Different applications use different data types according to their specific needs.
- Incompatible data structures often result because of the different groupings of data elements by software applications. For example, an element with a common logical definition in two applications may still be stored with two different physical representations.
- Applications written in different languages sometimes interpret their data values differently. For example C and COBOL interpret binary numeric data values differently.

What is needed, therefore, is an off-the-shelf product that is specifically designed to interconnect applications that were not originally designed to work together. That product must automatically, quickly, efficiently, and cost-effectively integrate applications having incompatible programming interfaces at the same or different functional levels of an organization. HP Enterprise Link is such a product.

HP Enterprise Link is an interactive point-and-click software product that is used to connect software applications (such as business planning and execution systems) to control supervisory systems found on the factory floor. HP Enterprise Link greatly reduces the cost and effort required to interconnect such systems while eliminating the need for custom software.

The Data Transfer Problem

The problem of transferring data from one software application to another is conceptually simple: just fetch the data from one system and place it in another. In practice the problem is more complex. The following issues arise when trying to implement electronic data transfer solutions:

- There must be a way to obtain data from the software application serving as the data source. Such access, for example, might be provided by a library of callable C functions.
- There must be a way to forward data to the software application serving as the data destination. For example, data might be placed in messages that are sent to the destination application.
- There must be a specification of exactly what to fetch from the source application and exactly where to place it in the destination application.
- The data being transferred must be translated from the format provided by the data source to the format required by the data destination.
- There must be a specification of the circumstances under which data should be transferred and a way to detect when these circumstances occur.

All of these issues are addressed in HP Enterprise Link.

HP Enterprise Link

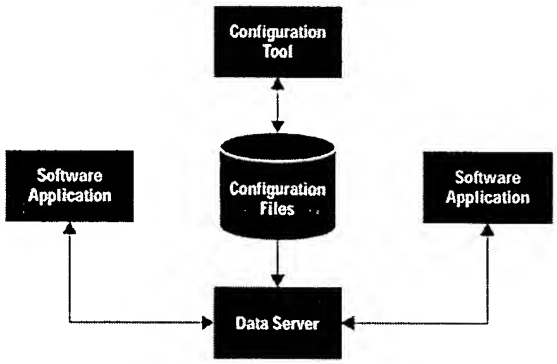
HP Enterprise Link product consists of the three components shown in **Figure 2**:

- An interactive configuration tool. This interactive window-based application allows users to direct the movement of data between two software applications.
- A data server. This noninteractive process runs in the background. It moves data in accordance with the directives that the user specified with the configuration tool.
- Configuration files. This is the set of mappings and trigger criteria created by users. The data is stored in configuration files. These files are created and modified by the configuration tool and read by the data server.

Linking Components

The HP Enterprise Link components listed above have the common goal of enabling users to create middleware that maps components with different interfaces together for data transfer.

Figure 2
The components of HP Enterprise Link.



In HP Enterprise Link, the combination of a single source address and a single destination address is called a *mapping*. A unit of data at the specified source address is said to be mapped to the specified destination address. In other words, it can be read from the specified source address and written to the specified destination address.

Although a mapping deals with the transfer of a single unit of data, real-world situations usually require the transfer of many units of data simultaneously. Therefore, HP Enterprise Link collects mappings into groups called methods. A method contains one or more mappings.

Mappings describe what to transfer and where to transfer it, whereas triggers describe exactly when to do the transfer. Data is actually transferred whenever a specified trigger condition is satisfied. This condition is called the trigger criterion. There are many possible trigger criteria such as:

- Whenever a unit of data at a specified source address changes value
- Whenever a unit of data at a specified source address is set to a specified value
- Whenever the source data becomes available—such as arriving in a message
- At a preset time of the day or a preset day of the week.

HP Enterprise Link considers trigger criteria to be part of the definition of a method. All the mappings for a single method share the same trigger criteria. Whenever the trigger criteria are met, HP Enterprise Link transfers—in unison—all the data specified by the method's mappings.

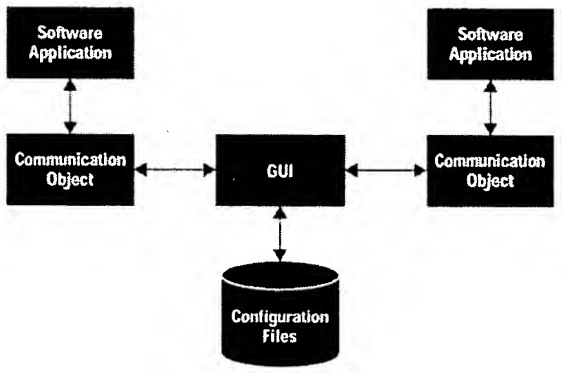
Multiple methods can simultaneously exist in HP Enterprise Link. For example, a user can create one method to transfer a particular production recipe from a business enterprise system down to a factory-floor control system. Conversely, raw-material consumption information for the recipe currently in production could be transferred periodically from the factory-floor control system up to the business enterprise system, using a second method.

The Configuration Tool

The HP Enterprise Link configuration tool provides users with a view of each software application's name space, and the tool graphically depicts what data to transfer and under what circumstances such transfers should occur (**Figure 3**).

The HP Enterprise Link configuration tool is composed of communication objects and a graphical user interface (GUI). Communication objects are used to obtain namespace data that is unique to each application and to provide application-specific windows. The configuration tool provides the user with an easy-to-use point-and-click style GUI.

Figure 3
The HP Enterprise Link configuration tool.



All dependencies on particular software applications are encapsulated in communication objects. The configuration tool's communication objects provide the following functionality:

- They fetch namespace information from communicating software applications for presentation to the user.
- They provide routines to create and manage application dependent control panel widgets, such as those used to specify triggers unique to a particular software application.
- They provide routines to tell the GUI exactly what functionality is supported by a communication object. For example, can the application software serve only as a data source (supply data values), or can it serve as both a data source and a data destination (both supply and use data values)?

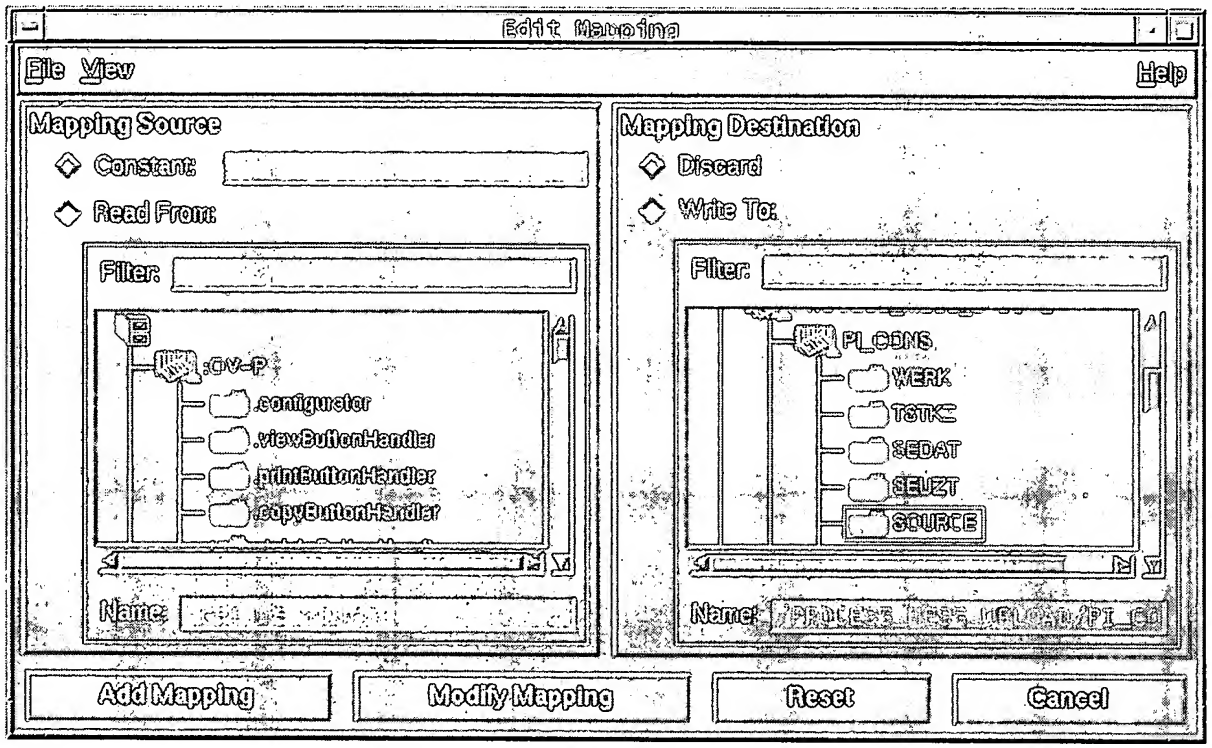
There are three important windows in the configuration tool's GUI: the Edit Method window, the Edit Mapping window, and the Trigger Configuration window.

Edit Mapping. The Edit Mapping window is used to create new mappings (**Figure 4**). The namespaces of both the source software application and the destination software application are shown. They are graphically displayed as tree diagrams. This makes it easy for users to specify which data to move where. They don't have to remember the names of data sources or data destinations. Instead they just choose from the displayed list of possibilities. The side-by-side display of application namespaces makes it much easier to integrate the applications.

Tree diagrams are used because they make large namespaces manageable. A linear namespace display was rejected early in the design of HP Enterprise Link because a flat list representation would only be manageable with software applications having a small namespace. Another advantage of tree diagrams is that most users are already familiar with them from file selector windows found in many software applications.

To create a new mapping the user selects an item from the Mapping Source tree diagram and an item from the Mapping Destination tree diagram, and then clicks the Add Mapping button. A new mapping is added to the mapping table displayed on the Edit Method window (**Figure 5**).

Figure 4
The Edit Mapping window.



Multiple static mappings can be created in a single step using branch assignments. This requires that the last component of the source and destination addresses be identical (so that appropriate mappings can be automatically created). Mappings can also be automatically created at the time methods are triggered. This is called dynamic mapping and requires the user to specify algorithms that can select source addresses and transform these addresses to valid destination addresses.

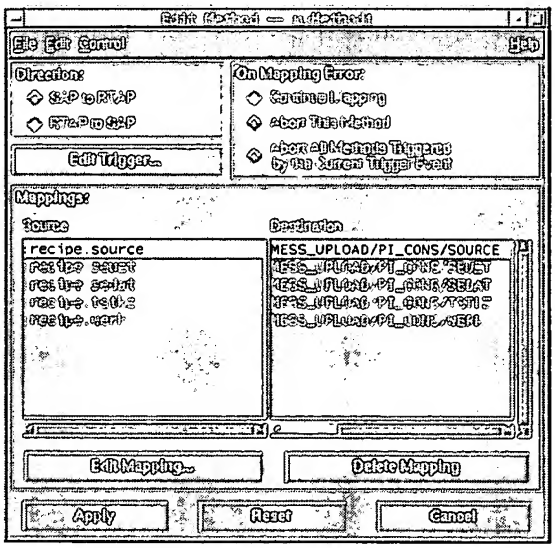
Edit Method. The Edit Method window (**Figure 5**) displays a method's mappings as a two-column table titled Mappings. Source addresses appear in the left column and destination addresses appear in the right. The data server transfers mapped data from source addresses to destination addresses in the same order as the mappings are listed in this table. The Mappings table makes mappings both explicit and intuitive to the user.

This window allows the user to specify in which direction to transfer data. All of a method's mappings specify data transfers in one direction—from one software application to another. The Edit Method window also allows the user to specify how to respond to errors that occur during data transfers. This will be described later in more detail.

Trigger Configuration. The Trigger Configuration window is used to define trigger criteria (**Figure 6**). This window displays all possible triggers to the user, as well as the currently configured trigger criteria. The Trigger Configuration window is designed to make setting up trigger criteria explicit and intuitive for the user.

The Trigger Configuration window is split into three groups: time triggers, triggers unique to the source application, and triggers unique to the destination application. Time triggers allow the user to specify that data mapping start

Figure 5
The Edit Method window.



at some specified time and repeat at a specified time interval, but be synchronized to a specified hour/minute/second of the day/hour/minute.

Triggers unique to the source application, such as the RTAP (real-time application platform) triggers shown in **Figure 6**, allow data to be mapped when something interesting happens in the source application. For the RTAP triggers in **Figure 6** interesting events include a database value change or the occurrence of an RTAP database alarm. Data can also be mapped when something interesting happens in the destination application.

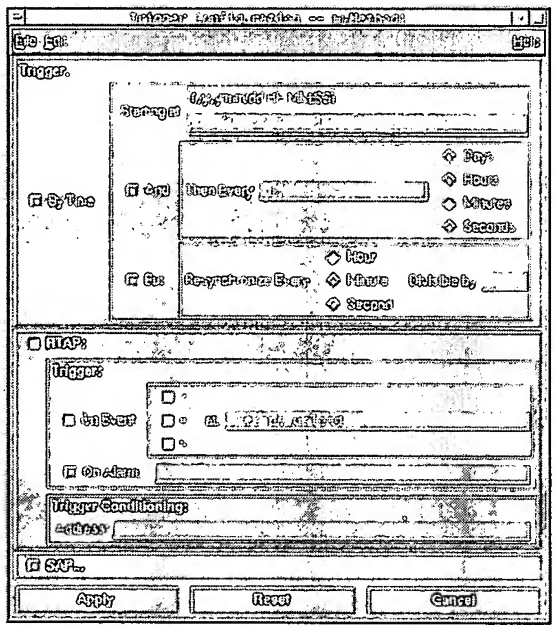
Thus, triggers allow data transfers to be pushed from the source application, pulled from the destination application, or scheduled by time.

Summary. Using the windows just described, users can create methods with the configuration tool. These methods specify one or more mappings and associated trigger criteria. This information is saved in one or more configuration files. The data server then reads these configuration files to implement the user's methods.

The Data Server

The HP Enterprise Link data server is composed of communication objects, a trigger manager, and a mapping

Figure 6
The Trigger Configuration window.



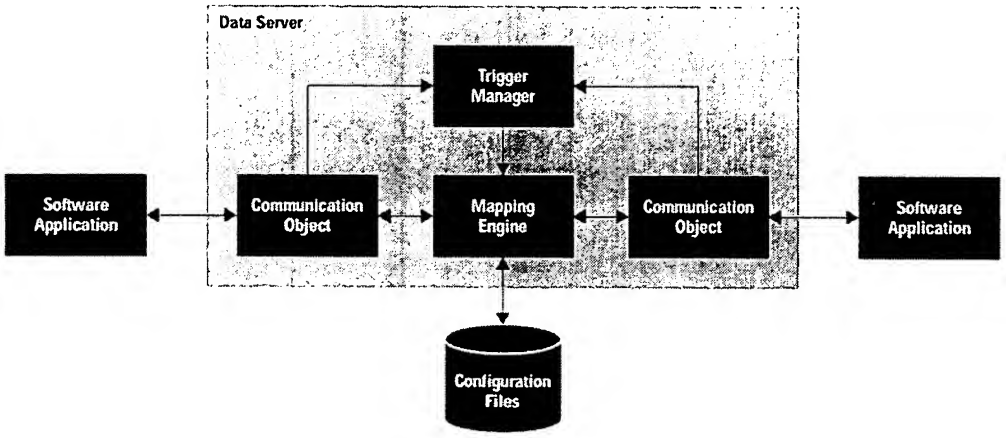
engine (**Figure 7**). Communication objects deal with the problems of generating triggers and getting data into and out of software applications. The trigger manager deals with dispersing Trigger Configuration data, coordinating trigger events, and notifying the mapping engine of trigger events. The mapping engine deals with the problems of reading configuration files, responding to triggers, mapping source addresses to destination addresses, and transforming the data as it is being mapped.

All software-application dependencies are encapsulated in communication objects. Communication objects serve as translators between external software applications and the data server's mapping engine—they translate the software application's native application program interface (API) to the interface used by the mapping engine.

The interface between a communication object and the mapping engine is standardized, with all communication objects using the same interface. For data that is being transferred, the interface consists solely of address-value pairs, where the address is from the application software's namespace, and the value is encoded in a neutral form. Thus a communication object only needs to be

Figure 7

The components of the HP Enterprise Link data server.



aware of its own namespace and how to convert between the software application's proprietary data formats and the neutral HP Enterprise Link data format. For triggers, the interface consists of well-documented interactions between the trigger manager and the communication objects.

Communication objects are usually distributed. They are split into two parts that are interconnected by a communication channel such as a TCP/IP socket. One part of the object is incorporated into the HP Enterprise Link data server process, while the other runs on the same machine as the corresponding software application. When a communication object is not split into two parts, the object, the data server, and the software application must run on the same machine.

Communication objects communicate with their corresponding software applications through whatever mechanism is available. For example, this could be through a serial port, shared memory, shared files, TCP/IP sockets, or an application program interface (API).

When a communication object transfers data, it translates data between the format used by the source software application and the neutral format required by the mapping engine. For example, for numeric values, a communication object may have to translate between binary IEEE-754 floating-point format and the mapping engine's neutral format.

In practice, not all data transfer attempts will be successful. For example, a particular source address might have been deleted, or a destination address may no longer exist. The configuration tool is used to specify what the mapping engine should do in this situation, and the data server must detect the condition and deal with it appropriately. When data transfer attempts fail, the user can have the data server do any one of the following:

- Continue mapping data (ignoring the error)
- Abort all subsequent mappings associated with the current method
- Abort all subsequent mappings and all subsequent methods triggered by the current trigger event (if multiple methods were simultaneously triggered).

The interface between the communication object and the mapping engine is designed to support transaction-oriented data transfers, using commit and rollback. This functionality comes into play when mapping attempts fail. It allows the data server to undo (roll back) all data transfers done in all currently processed mappings associated with the method's current trigger event.

The Running Data Server

When the HP Enterprise Link data server starts up, it reads the configuration files that the user created with the configuration tool. It then prepares to deal with the specified

trigger criteria, usually by notifying the appropriate communication object to detect it. Finally, it enters an event-driven mode, waiting for the trigger criteria of any configured method to be satisfied.

When either a source or destination communication object in the data server detects that a method's trigger criteria have been satisfied, the object informs the data server trigger manager that a method has been triggered. This starts the mapping engine. Alternatively, if the data server trigger manager detects that a method's time-based trigger criteria have been satisfied, the mapping engine starts.

When triggered, the mapping engine requests that the source communication object provide the current data values at the method's configured source addresses. The source communication object obtains these values from the software application, translates the format of all fetched data values to a neutral format, and passes the result to the mapping engine as address-value pairs, with one such pair for each of the method's defined mappings.

The data server mapping engine looks up the destination address that corresponds to each source address. This lookup results in a new list of address-value pairs, with the address now being the destination address, and the value unchanged (and still expressed in the mapping engine's neutral format). To minimize the impact on performance, this lookup is implemented using a hash table.

The mapping engine sends the new list of address-value pairs to the destination communication object. The destination communication object converts the received values into the format required by the destination software application, and writes the converted result to the specified addresses in the destination software application.

Communication Objects and Software Applications

There are two fundamental ways for software applications to provide communication objects access to their data: the *request-reply* method and the *spontaneous-message* method.

In the request-reply method, the communication object sends a software application the address of a wanted data unit in a request and receives its current value in a reply. With this method the communication object controls the data transfer. It determines which unit of data to read and when to read it. Structured Query Language (SQL) and real-time databases are two examples of software applications that employ the request-reply method.

In the spontaneous-message method, communication objects receive data, usually as messages, from the software application whenever the application chooses to send it. With this method the software application controls the data transfer. It determines which data to provide and when to provide it. SAP's R/3 product is an example of a software application using the spontaneous-message method.

The method that a software application employs to provide external data access determines the trigger criteria that are possible for that application's communication object. The request-reply method allows event, value, and time-based trigger criteria since the communication object controls the data transfer. The spontaneous message method is limited to value-based triggering (essentially filtering) because the software application providing the data controls the data transfer.

Spooling

The HP Enterprise Link data server's communication objects must cope with communication failures. This means that outgoing data must be locally buffered until a communication object verifies that the application software, when acting as a destination, has successfully received it. It also means that incoming data must either be safely transferred through the mapping engine or locally buffered when a communication object accepts data from the source application software.

Spooling is especially important if the source application is separated from the HP Enterprise Link data server by a wide area network (WAN). WANs are considerably less reliable than local area networks, and thus are more likely to lose data.

In a typical HP Enterprise Link installation the data server runs on a machine located near or on the factory floor. Production orders are downloaded from the enterprise level to HP Enterprise Link as soon as they are available. The downloaded data is buffered at the factory until it is needed. Using HP Enterprise Link in this way reduces the probability that the factory would lack unprocessed production orders if the WAN is down for a prolonged period of time.

Buffered data must be preserved even if the HP Enterprise Link host machine is shut down or crashes. To do this, HP Enterprise Link stores buffered data in disk-resident spool files.

The amount of storage used to hold buffered data must be restricted to protect the host computer from failure caused by insufficient resources. HP Enterprise Link can limit the size of spool files by controlling:

- The maximum size of spool storage
- The maximum number of messages buffered
- The age of the oldest message buffered.

The user can set any one or all of these limits, using the HP Enterprise Link configuration tool.

Tracing

HP Enterprise Link allows the data being transferred to be monitored by the user. The monitoring is called *tracing*. Tracing is useful for creating an audit trail of the transferred data and for debugging and testing methods. Tracing does not affect the data being transferred.

The configuration tool is used to enable and disable tracing, but it is the data server that generates trace messages when tracing is enabled.

Data can be traced at a number of different internal locations within the data server (see **Figure 8**). Some of the forms in which trace results can be expressed include:

- Data as received by a data server communication object from a source software application. This trace data is expressed using the source software application's native data format and includes the source address, the value received or read, and the time of transfer.
- Data as sent by a data server communication object to the destination software application. This trace data is expressed using the destination software application's native data format and includes the destination address, the value sent or written, and the time of transfer.
- Data being mapped by the mapping engine. This trace data is expressed using the data server mapping engine's neutral data format and includes the source address, the destination address, the value transferred, and the time of transfer.

Error messages reported by the mapping engine or by communication objects can also be included in the trace output. This ability ensures that the relative sequencing of data transfer messages and error messages is preserved, which greatly aids the user when trying to troubleshoot mapping problems.

Server Data Flow

HP Enterprise Link allows the flow of data in the data server to be interrupted at a number of different internal points (see **Figure 9**). This is useful for isolating the effects of data mappings during debugging and testing. When an information flow is interrupted, data does not pass the point of interruption; instead, the data is discarded.

The flow of information being transferred from a communication object to a software application can be interrupted. Interrupting the flow here allows the data server

Figure 8

Tracing data that is transferred between applications.

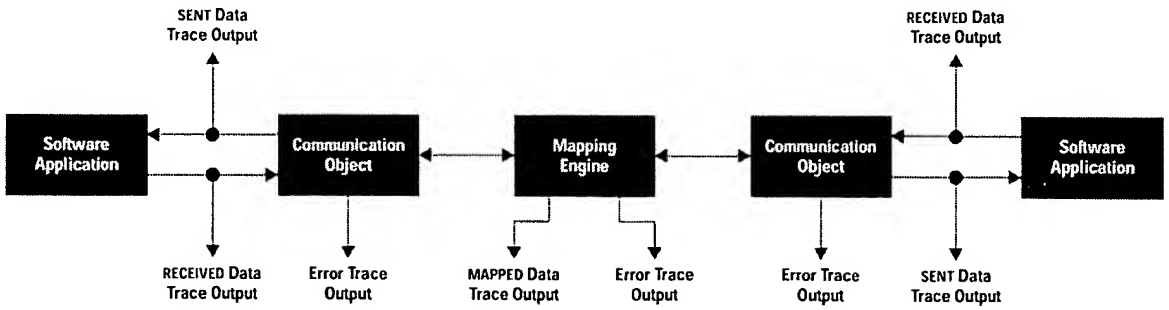
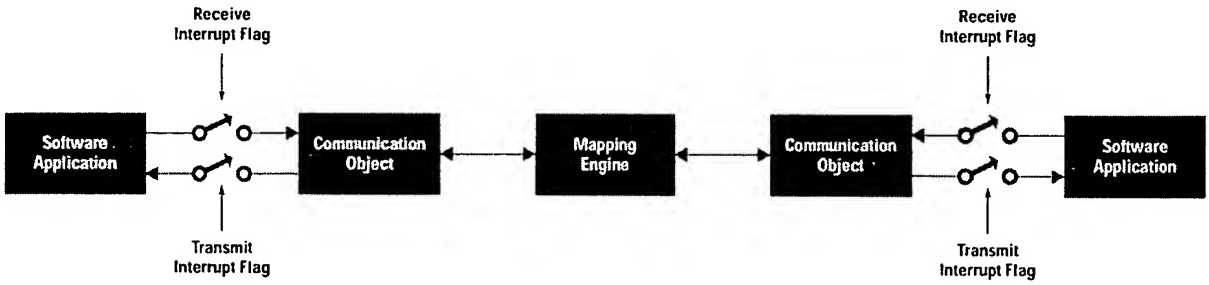


Figure 9
Interrupt locations in the data server.



to read from mapped source addresses, map to new destination addresses, and then discard the data just before it would have been written to the destination software application.

The flow of information being transferred from a software application to a communication object can also be independently interrupted. Interrupting the flow here allows the data server to ignore all data sent to the communication object by the source software application.

Data Integrity

The HP Enterprise Link data server is carefully designed to preserve the integrity of the data being mapped and to map the data exactly once for each trigger event. The design was influenced by considering how to react to communication channel failures and data server process terminations. The circumstances that could cause the data server process to terminate are the following:

- If a person or software process explicitly kills the data server process
- If the host machine suffers a hardware or software failure, loses power, or is manually turned off.

Communication channel failures must be handled carefully. If the communication channel connecting a communication object to its software application fails, the data being mapped at the time of failure must not be lost or duplicated. Also, after normal operation of the communication channel is restored, communication between the communication object and its application must be automatically established again and all interrupted data transfers restarted.

The following steps are taken to ensure data integrity when communication channels fail:

- For data received from the source software application, the communication object never acknowledges receipt of the data until the data has safely been saved to a disk-resident receive-spool file.
- Data received by the communication object from the source software application is not removed from the receive-spool file until the data has successfully passed through the mapping engine and been forwarded to the communication object responsible for sending it to the destination software application.
- The communication object that sends data to the destination software application only notifies the mapping engine that it successfully received the data after the data has been safely saved to a disk-resident transmit-spool file. Also, it only removes data from the transmit-spool file when the destination software application has acknowledged successful receipt of the data.

Conclusion

The HP Enterprise Link product greatly reduces the cost and effort required to interconnect business management systems (such as SAP's R/3 product) and measurement and control systems (such as Hewlett-Packard's RTAP/Plus product). HP Enterprise Link is an off-the-shelf product that allows users to connect software applications using an easy-to-use point and click graphical user interface.

With HP Enterprise Link, companies can minimize the costs associated with changes made to computer systems and adapt more quickly to changing market conditions.

Acknowledgments

The author wishes to thank Andrew Ginter and Andy Mah for their significant contributions to the design and development of the HP Enterprise Link product. John Burnell for his comments during the design of the product, and Steve Heckbert for his valuable feedback.

Online Information

For more information about HP Enterprise Link, take a look at the information located at the following URLs:

- http://www.tmo.hp.com/tmo/pia/Vantera/Index/English/Index.html
- http://www.tmo.hp.com/tmo/pia/Vantera/Index/English/Products.html

▶ Go to Next Article

▶ Go to Journal Home Page

40/3,K/87 (Item 9 from file: 148) **Links**
Gale Group Trade & Industry DB

08410982 **Supplier Number:** 17823355 (USE FORMAT 7 OR 9 FOR FULL TEXT)
PC graphics reach new level: 3D.(PC 3-D graphics architecture, techniques, terminology)(includes related article on 3D graphics glossary) (Technology Information)

Yao, Yong
Microprocessor Report , v10 , n1 , p14(6)
Jan 22 , 1996
ISSN: 0899-9341
Language: English
Record Type: Fulltext
Word Count: 4702 **Line Count:** 00384

...and rendering. In the tessellation stage, a description of an object is created and the **object** is then **converted** to a set of polygons. The geometry stage includes transformation, lighting, and setup. The rendering...

...maximizing performance.

In the tessellation stage, a description of an object is created, and the **object** is **converted** to a set of polygons. Then, the geometry stage performs the following tasks:

Transformation: Translating and rotating 3D **objects**, thereby positioning the viewer within the 3D space.

Lighting: Determining lighting, shading, and texture characteristics ...the U direction will change. If the addition of a du or a dv does **not** cause the UV address to **change**, the same texel is **mapped** onto the triangle; if a change is made, a new texel is mapped onto the...

...written to the frame buffer. The disadvantage is the blocky image that results when the **object** moves. More advanced texture-**mapping** algorithms include bilinear filtering and MIP **mapping.**

Z-buffering. When **objects** are rendered into a 2D frame buffer, the rendering engine must remove hidden surfaces. The...image. It takes into account the effect of the Z value in a scene while **mapping** texels onto the surface of polygons. As a 3D **object** goes away from the viewer, the length and height of the object become compressed, making ...

...purpose of MIP mapping is to remove this effect.

This algorithm stores a number of **different** sizes of the texture **map** in memory, representing **different** resolutions. When a 3D **object** is large, due to its proximity to the viewer, a correspondingly large texture **map** is used. As the **object** moves away from the view point, the rendering engine switches to a smaller texture size...

...in which the four sampled texels are fetched from one of several versions of the **original** texture **map**.

Since the number of texture maps is limited, one artifact possible in a MIP-**mapping** approach occurs when an **object** is moving toward or away from the view point. As the **object** crosses a MIP boundary, a **change** can sometimes be noticed when switching from one texture **map** to **another**.

Trilinear MIP **mapping** solves this problem by interpolating between MIP map levels. The texel value is computed by...

...are written to the frame buffer. The texture is modulated with respect to the lighting **source**(s). Lighted texture **mapping** is a feature beyond Gouraud shading with an intensity ramp to achieve lighting effects.

3D...

...the GUI model increased the complexity of both software and hardware compared with simple text-**mapped displays**, moving to 3D adds **another** layer of complexity. Yet with increasing transistor budgets and the availability of inexpensive DRAM and...

...these concepts are only summarized here.

Alpha-blending--Technique for adding transparency information for translucent **objects**.

Alpha buffer--An **extra** channel to hold transparency information; pixels become quad values (RGBA).

Anti-aliasing--Subpixel interpolation, which...Parvum (Latin) means many in one. A method of increasing the quality of a texture **map** by applying **different**-resolution texture **maps** for **different objects** in the same image, depending on their size and depth.

OpenGL--Silicon Graphics' high-level...

...pixels.

RealityLab--Microsoft's high-level object-based 3D API, originated by Rendermorphics. It directly **manipulates objects**, lights, and cameras.

Rendering--The stage of the 3D pipeline that creates a 2D display...

...of rendering based on subpixel interpolation.

Tessellation--The first stage of the 3D pipeline, which **converts** a description of an **object** into a set of three-dimensional polygons.

Texture filtering--Removing aliasing artifacts, such as sparkles and blockiness, through interpolation of stored texture images.

Texture **mapping**--Wrapping textures around **objects** to add realism or to reduce complexity.

Transformation--Translation and rotation in a 3D environment...

19960122

40/3,K/32 (Item 3 from file: 16) **Links**
Gale Group PROMT(R)

05197161 **Supplier Number:** 47930081 **(USE FORMAT 7 FOR FULLTEXT)**

Challengers vs. Champ: We test two up-and-coming device management packages against the champ of the EMS arena II
CommunicationsWeek , p 43
August 25 , 1997
Language: English **Record Type:** Fulltext
Document Type: Newsletter ; Trade
Word Count: 3508

...define custom alarm conditions. When an alarm occurs the Network Manager Console beeps, the network **map displays** an alarm condition and the event is logged into an alarm database. Often alerts went...

...between uses. Time and frustration could be saved if OpenView Professional automatically opened the last **map** each time it started. Many of the **list** boxes do not allow multiple selection. Another frustrating limitation was that the device access **list** was limited to each local **map**. This prevents any way to specify global defaults.

The map drawing tools are somewhat primitive as well. For example, there was **no** way to **change** the symbol of a **mapped** device without **first** deleting it and then adding a new symbol. Also, the selection of symbols was limited. Furthermore, mapping tools did not support sub-net designations, or the topology definition. There was **no** way to rearrange or center **maps** on the **display**. On a **map** there was no way to distinguish between similar devices--all routers looked the same, all...Cisco routers. It provided more information on the Cisco router than the 3Com.

TNG offers **alternates** to **Map** views that also are useful, such as a node **listing** called ObjectView. Similar to Windows Explorer with nested directories, ObjectView shows any additional information known...

...to the network, discovery must be restarted. Then, after the nodes are added to the **database**, the **map** or topology browser must be manually refreshed. Nodes that are disconnected are reported, and alerts can be configured. The status change appears in the **map display** and in the alerts **log**, with the type of alert color-coded.

The standard Unicenter TNG can do some reconfiguring of selected routers and switches. As part of **ObjectView**, the **configuration** can be **changed** by re-entering the settings if the configurations are exposed in the management information base...

...daily basis.

TNG is visually appealing. It's easy to distinguish device types from

their **map** symbols, and the **list** of available symbols is extensive.

Unicenter TNG has a Design Mode that lets the user...add-in options. To manage a 100-node network with agents on five servers, the **list** price including the **first** year maintenance is about $37,000.

To manage a 250-node network with agents on 10 servers, the **list** price including the **first** year's maintenance is about $62,000. This translates to $250 to $370 per node...

...using a combination of temporary leased equipment and LANQuest Labs' internal company network.

After our **initial records** of each product's ease of installation, as well as how many additional applications it...

19970825

REVIEWS

August 25, 1997

Challengers vs. Champ

By GAIL JAMES and MICHAEL SPIGELMIRE

What does it cost per user for your company to maintain your corporate data network each year? Is it $2,000, $5,000, $10,000--more? How many users and servers can one network manager support? Is it 200, 100, 50 or fewer? It's clear today that the only way to restrain cost-of-ownership for complex distributed enterprise networks is to invest in network management tools and training so managers can do their jobs more efficiently. The results can be dramatic.

Some organizations have been successful in cutting cost-of-ownership by over 50 percent--enabling their network managers to double the number of users and servers supported.

Recent market studies by International Data Corp. and other analysts indicate that successfully implemented enterprise management systems can provide annual savings of $1,000 to $5,000 per user. If your company network has 500 or more nodes and the cost to implement an enterprise management system is $75,000 to $100,000, that amounts to less than $200 per user. The cost savings make the decision a "no brainer." But, if your network is only 100 t o 200 nodes, you might think twice before spending $500 per node.

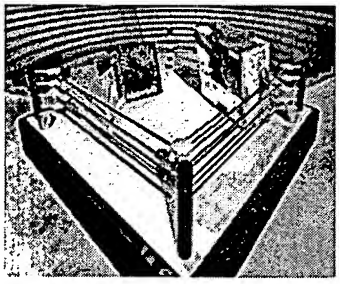
For managers in this position, there are alternatives. Network management systems (NMS) for small-to-medium-sized networks can be implemented for under $20,000.

To help you make informed decisions about which network management platform best meets your needs, we set up a challenge match pitting two leading PC LAN-oriented network management systems, HP OpenView Professional and Novell's ManageWise, against Computer Associates International's Unicenter TNG, the steady leader of enterprise management system solutions. We wanted to find out how far we could take our middleweight contenders, and when it pays to bite the bullet and spring for the heavy guns.

This NMS challenge took place in July using a test network composed of routers, switches, multiple LAN segments, WAN connections, several network servers and over 60 workstations and PCs. Seven network management systems vendors, some enterprise heavyweights and some PC LAN middleweights were invited.

SUMMARY

Can leading network management packages outpunch CA Unicenter TNG for small- to-medium-size networks?



Links to related stories:

- How We Tested
- RMON Rocks On
- Heavy Weight Wins TKO

Links to previous Reviews:

Testing ATM In The Real World

Firewall Face-Off: Eagle and Gauntlet

Windows: '97? '98? '99?

Of the heavyweights, only Computer Associates (CA) decided to climb into the ring. Though we asked Hewlett-Packard to participate on an enterprise level (something OpenView is capable of in certain versions), HP was only able to supply us with the smaller, Professional Suite version. Vendors who declined the challenge included IBM/Tivoli, Sun Microsystems, Cabletron and Platinum Technologies.

Overall, we were impressed by all the contenders, although in different ways. We not only reviewed each product as a whole, but also compared all three using specific criteria. For example, when it came to something as basic as installation, we found ManageWise to be straightforward and reliable, while the other two had some problems. Unicenter TNG is a big product with lots of installation options, meaning it took awhile to get everything properly configured. OpenView Professional should have had an easier install process as well, but in fairness we were working with a beta version of the new OpenView and not a fully shipping product. ManageWise only took a hit for some problems with the Network Manager Console portion of the installation.

Our second criteria was very important, carrying double weight in the final scoring. The power of any network management system starts with its ability to find and manage existing network equipment. TNG was the dominator, finding most of our test network immediately and the rest with only a little extra time and guidance; it also recognized the routed sub-nets in the process. ManageWise and OpenView Professional wound up doing only half the job. ManageWise found our IP/IPX nodes quickly, but failed to find most IP-only nodes. OpenView was just the reverse. It did a good job with IP-only nodes, but couldn't find any IP/IPX nodes. TNG out scored the efforts of the other two combined.

Overall, we had 10 product criteria to judge our contenders. Aside from the two mentioned, we also looked at application management, traffic and performance metering, troubleshooting and fault management, console look and feel, management reports, flexibility and third-party support, pricing, and even vendor support and commitment.

While you can read how each product performed in the reviews, some criteria requires further discussion. For example, traffic and performance metering is a very important category for any network management solution. For traffic management, TNG is a good listener and reporter as is ManageWise, if the language is IPX. TNG gathers SNMP and RMON data, but we were disappointed to find it doesn't measure performance or provide trend analysis.

ManageWise has a third-party protocol analysis product built-in, called LANalyzer, which provides good utilization and performance data, but only on its local segment. OpenView has an NT version of HP's NetMetrix traffic performance analyzer, but this beta version was unstable and had to be removed. We were surprised to conclude that all three contenders need t o be more proactive to get a handle on real-time performance and analysis.

Another key factor for network and IS personnel is the flexibility with which they can access and view their network's management reports. While it was no revelation to discover TNG's reporting was fairly complete and understandable, we found ManageWise to be not as thorough but still a good provider of server and alarm tracking reports. OpenView, on the other hand, surprised us by providing only viewing--reports can't be sent to a file or printer.

The criteria dealing with third-party and even internal vendor support also needs explanation. Implementing one of these solutions in a network of any size means a substantial commitment on the part of the customer, so it's only logical buyers would want to be certain of their chosen product's popularity and life expectancy. And because all three contenders also act as platforms for running third-party add-on software, gauging how their NMS is seen in the eyes of the industry also is important.

In the case of CA's TNG, this product's breadth and name recognition are paying off. CA is obviously extremely committed to the product since, by all accounts, it makes up around a quarter of the company's total revenue.

The company announced recently its intention to beef up the software with new automatic agent features and Internet management capabilities late in 1997. It also recently shipped Unicenter/Advanced Help Desk (AHD), which provides a Web interface and knowledge tool

First Network-Ready PCs Prove Their Net Worth

Verity IntelliServ 1.0: Coping with Info Overload

For additional stories:

Hands On

Max Award

First Look

capabilities. As one of the market leaders in enterprise management software, TNG offers excellent flexibility and third-party support.

By contrast, analysts estimate that HP's OpenView platform, as a whole, carries almost 16 percent of the NMS market, and has generated substantial third-party interest as well. And it's not just in selling add-on software functionality, but also on the hardware side, where droves of vendors are ensuring that their products can be man aged by OpenView. Managers implementing OpenView can rely on the support of over 500 third-party plug-ins that allow OpenView to control a wide variety of network resources.

Yes, it's true that the base Professional package we reviewed here shows OpenView in somewhat of a pale light next to behemoth TNG, but remember we didn't test add-on software.

There was simply too big a list to choose from, and HP decided not to choose for us. And remember we also didn't get our hands on the full-blown enterprise version. HP has announced and delivered significant enhancements to OpenView Network Node Manager, even during our testing process, including a new Java-enabled console and extensions to the bundled HP NetMetrix metering tool that will allow managers to track network response time.

ManageWise proved that Novell still carries some weight--but we found it needs a new OS platform to improve flexibility and get third-party vendors re-energized. Novell is responding to some of th ese needs with its upcoming Java-based version, code-named Houston. According to Novell, this version will allow network administrators to manage their networks via any Java-enabled Web browser and is designed to deliver a distributed model of network management.

The company has also released two new tools for the Windows NT platform. Workstation Manager lets users manage Windows NT workstations using NDS, while Novell Administrator for Windows NT integrates NDS tree structures with Windows NT domains.

Novell is forging key industry alliances that will secure ManageWise's long-term future. Novell and CA have announced multiple partnerships, covering not only the ManageWise version that will integrate with the larger Unicenter TNG framework, but also an agreement that may lead to the integration of Novell's NDS directory technology into TNG.

For now, the deal means that Unicenter users will receive a special bundled version of ManageWise for managing their IntranetWare segments, and Mana geWise users will find copies of CA's ARCserve backup software. The companies plan for a much deeper integration in the future, including seamless moving of network objects between the two systems and using NDS to allow Unicenter much greater power as a central network information repository.

And Novell is still strongly encouraging other third-party vendors to build add-on functionality for the basic ManageWise platform. One example is NetPro Computing Inc.'s Config Central for NetWare, a new ManageWise snap-in that handles NetWare Loadable Module (NLM) software management.

For the future, each vendor claims to strongly support its respective combatant. CA has the most at stake and is investing the heaviest. HP is emphasizing OpenView Network Node Manager, but OpenView Professional doesn't seem to be taking off among users as well as expected. Novell has been diverted with other concerns, but is now investing in ManageWise's future.

Although each product had a few surpris es, good and bad, they fit their reported niches very well. TNG is still the best choice for those managing large installations needing as much as they can out of one NMS solution. ManageWise is still the top tool for dedicated IntranetWare shops, but is expanding fast to be able to handle a medium-sized network with heterogenous intranet operating systems as well.

HP's OpenView was still the darkest horse in the challenge even after testing, mainly since

we were limited to the OpenView Professional product, which is a subset of the larger OpenView Network Node Manager. Even so, the Professional suite showed promise for small-to-medium-sized PC-oriented LANs.

Novell's ManageWise 2.1

For this test, a Compaq Prolinea 5133 (Pentium 133) with 32 megabytes of RAM was used as the server running NetWare 3.12. To use the Network Explorer feature of ManageWise, a minimum of 32 megabytes is required. The server installation was smooth. For the Managers' Console, the Windows 95 installation failed three times. However, the version for Windows for Workgroup 3.1 installed without problems. Back on the server, the ManageWise installation proceeded automatically and created numerous files on both the server and Manager's Console. The post install configuration tasks were straightforward and well documented.

One nice feature is that ManageWise doesn't care what version of IPX services you're using. You can be running NetWare 3.12, NetWare 4.1 or IntranetWare--it's all the same. The only limitation is that the current version does not recognize NDS services.

After installation, we delved into the ManageWise server configuration screen. The icons allowed the network manager to launch each service shown in a straight forward manner. By clicking on the connections icon, a quick table of connected nodes was provided. Once we started things rolling, ManageWise began the discovery process using IP. IPX discovery was enabled via LANalyzer, the built-in software protocol anal yzer tool.

Overall, the IPX nodes were accurately found and quickly mapped, but our test network's IP nodes were inaccurately discovered and this process took longer, too. IPX discovery included a second NetWare server and a LaserJet print server, displaying its MAC address.

When the IPX and IP discovery process was complete, only four of the seven routers were discovered and the switch wasn't found, either. Curiously, it listed a remote ISDN router as an unknown segment, apparently finding it in the centralized Cisco 7000's routing table. It failed to discover most of the IP servers and workstations. Once the auto discovery process was completed, ManageWise displayed as much of the network as it had found in a network map.

Alternately, we could have displayed an inventory list.

ManageWise has strong management capabilities for IPX nodes. When an IPX client logged onto the ManageWise server for the first time, it was automatically configured with all the files needed to monitor that client's network activity and resources. When this was complete, ManageWise could monitor and control the IPX clients extensively, including displaying each client's screen, remotely controlling mouse and keyboard and even rebooting the machine.

ManageWise displayed a small icon on the client screen and sounded a beep periodically at the client station to notify the user that it was being monitored. The ManageWise console also displayed each IPX client's hardware configuration, CMOS map, memory map, IPX version, installed software and packet statistics.

We also were a little disappointed at the way ManageWise managed the network map in terms of ongoing adds and changes. For example, the addition of new IPX clients happened quickly but new IP stations never appeared on the map. And, a removed IPX client was still on the map, without change, 45 minutes later. There was also no capability to access the routers or the switch for configuration information.

ManageWise does support some level of application management. For IPX nodes with the ManageWise client, the NetWare Application Launcher allows the managers to transfer files from server to client or client to server from the Manager's Console. This, in combination with remote control, means you can remotely load, install, configure and verify virtually any software installation at the client. If a ManageWise client was not currently logged onto the ManageWise server, that node was displayed on the map as an IPX icon with only the MAC address as identification.

When the node logged in, the Manager's Console showed the user's login name. Monitoring of non-ManageWise clients is not supported. Unfortunately, ManageWise does not provide application usage metering or monitoring capabilities beyond the use of client remote control.

As far as traffic and performance management goes, within the ManageWise server segment there was considerable traffic monitoring as LANalyzer is part of the server and con sole installation process. This includes a dashboard showing current traffic rate in packets per second (PPS), the percentage of bandwidth used on that particular LAN segment, error count shown as the number of packets that had errors in transmission, traffic trend graphing, packet capture, conversation monitoring (node to node) and several other capabilities. Thresholds can be set for utilization, errors and more. This capability is limited to LAN segments where the ManageWise server is located.

There were 19 servers monitoring options. But, only one item at a time was viewable since our console's OS was limited to only Windows 3.1. Viewable options included CPU utilization, available hard disk space, login count, disk reads and writes, and others.

Under our Windows 3.1 console setup, network managers can monitor multiple servers, but display information only one at a time, and there can be only one desktop inventory server. We didn't find any security management features, access violation reporting or user change capabilities in our test copy of ManageWise, but there was an inventory database and SNMP support for IPX, though not for IP.

ManageWise did include a standard virus detection feature and an alarm status display for all IPX clients including PCs, servers and routers. However, this did not extend to IP-only nodes--IPX also had to be running. We would also like to have seen fault tracking when the system sensed an alarm condition.

For day-to-day usability, however, we found ManageWise's displays to be straightforward, understandable and intuitive. It's easy to find the available information. But Windows 3.1 allowed only one display at a time. Presumably, the Windows 95 version supports multiple displays. Though Novell has announced both a Windows NT Workstation Manager and the Novell Administrator for Windows NT, neither was included with the test copy of ManageWise and were not available for our test.

The ManageWise server configuration window allows printi ng of most options, such as the Alarm Disposition Table. But, while the Client View retrieves extensive information about a managed client, the data can only be viewed, not directed to a printer or file.

For those looking to extend the functionality of their ManageWise systems, Novell is working hard to maximize this platform. Aside from vendor alliances, Novell is developing Java and Web-enabled management interfaces as well as integration with BorderWare. The company is still encouraging third-party development for additional "snap-in" functionality for ManageWise users. Many of these third-party applications exist through the Novell distribution channel, and include Alexander SPK, a server crash prevention and diagnostic tool from Alexander LAN Inc., Nashua, N.H; AlertPage from Geneva Software, Northbrook, Ill., for converting SNMP alerts into audible alarms; or SyncComplete from Kansmen Corp., Milpitas, Calif., which gathers essential network data into a readily accessible database.

ManageWise pri cing is based on the number of user nodes and the number of NetWare or Windows NT servers managed. Base pricing starts at five users. The license fee for a typical 100 IP-user network, with five NetWare or NT servers, would be just under $7,000. The license fee for a typical 250 IPX-user network, with 10 NetWare or NT servers, would be just under $13,000. The time saved in software distribution and PC configuration management alone would easily justify that cost.

ManageWise is a solid NMS for IPX network segments, but very limited for managing IP-only segments. Though Novell in general has long had a less than stellar reputation with regard to TCP/IP, we were a little surprised that even this latest version of ManageWise had done so little to address these difficulties. However, ManageWise is a very good choice for small-to-midsized corporate networks of 250 or fewer nodes, with IPX and mixed IP/IPX nodes. The timesaving benefits should provide a rapid payback on the investment in ManageWise lice nses and training.

HP OpenView Professional

The OpenView Professional version D (beta release) was installed on an HP Vectra 500 P5-166 with 32 megabytes of RAM, a 1.5-gigabyte IDE hard drive, and running Microsoft Windows NT 4.0 Server with Service Pack 3 installed. All actions were performed as the Administrator, no other user accounts were created.

Initially, the OpenView Professional suite was installed on Windows NT Workstation and performed poorly. The machine was then upgraded to NT Server. However, the discovery function was not identifying SNMP devices. Only after installing NT Server Service Pack 3 did the NMS begin identifying the devices. However, this was probably a Windows NT bug that Service Pack 3 fixed instead of an OpenView Professional problem.

Neither of these requirements were documented. The balance of the installation was simple. The CD-ROM documentation also indicated that Windows 95 was supported.

HP states that OpenView int egrates 13 management applications into one environment. However, most of the plug-in options, and all the bundled software, are 16-bit apps. When any of these crash, the entire OpenView process must be restarted. This occurred with regularity on our test network. Restarting also meant enduring the lengthy process of re-discovering the network nodes, since OpenView Professional does not save a last discovery map for later use or update. After installation, the start menu listed both McAfee's Saber LAN Workstation and Symantec's Norton Administrator even though these apps had not been installed.

To initiate discovery you must launch the AutoStart feature. This is designed to discover, map and poll all nodes in one step. OpenView Professional polling does not support IPX nodes, only IP nodes, just like its big sister Unix product OpenView Network Node Manager. We couldn't find any third-party add-ons supporting IPX. The test network was a combination of IP-only segments and dual IP/IPX segments.

Unfortunately, OpenView Professional was unable to monitor the nodes on the test network. It only discovered nodes on three of the seven LAN segments tested. Combination IP/IPX protocol nodes appeared to confuse it. It discovered only one of the two NetWare IP/IPX servers and no IP/IPX workstations. OpenView Professional did not recognize the one NetWare server as a NetWare server, until the Discovery Manager Dialog box was first opened and given the appropriate IP address.

For network mapping OpenView Professional cannot group networks with different net masks. This means not all segments could be connected and displayed in a single map.

But, it will show the relationship between different LAN segments as specified by the network manager. Also, OpenView Professional will not arrange nodes in a map according to physical segments. It's up to the manager to use the integrated drawing tools to make connections between nodes. OpenView Professional could not distinguish between si ngle IP address nodes and nodes with multiple IP addresses (such as servers with multiple network adapter cards). OpenView Professional found the networked HP printer, but curiously, the JetAdmin tool running alone never found the printer. To complete the map, JetAdmin had to be given the printer's IP address. OpenView Professional does not automatically poll nodes for status information, rather it polls only nodes specified by the Network Manager.

New nodes added to the network are only found by restarting the discovery process. When discovery is finished, the new nodes are specified on the map. When OpenView Professional is restarted, polling must be manually restarted.

OpenView Professional detected node disconnects, as long as workstations had been configured for polling. The alert is a single beep or one beep every few minutes. If down, the node symbol and all parents will turn red. Alerts can also be sent to another OpenView Professional Manager's Console. When used in conjunction with the optional feature applet, Notify! alerts can be sent to a pager (this was not tested). In one instance, a node went from a "known" device to "unknown," without any alarm being generated.

According to HP documentation, OpenView Professional has the power to manage and configure HP routers, switches and hubs (this was not tested). It was not able to configure the Cisco and 3Com routers or the Bay Networks switch on the test network, because even though these companies write support software for the large OpenView Network Node Manager, this software isn't supported under the Professional version.

For software distribution OpenView Professional relies on third-party software. It comes with evaluation versions of McAfee Associates Inc.'s Saber LAN Workstation and Symantec Corp.'s Norton Desklock Administrator. The evaluation version of Saber LAN provided on the CD-ROM could not be installed in a Windows NT environment. It was a NetWare version and, as stated previously, OpenView P rofessional did not work in the NetWare environment. McAfee's Web page description of Saber LAN Workstation features automating and metering software distribution is impressive. It's unfortunate it could not be evaluated.

The OpenView Expose software (licensed from Symantec) was supposed to discover and manage all NT and NetWare servers on the network, but this feature also failed to work and couldn't even find its own client on the same machine. Though this sounds bad, we're certain this particular error was the result of our having to look at Professional in its beta incarnation. This problem should be gone in the shipping release.

The evaluation version of Norton Administrator was installed; however, the distribution component would not work on the Network Manager's Console. The Norton Administrator Distribution Agent on a managed node could not find the OpenView database on the server. The inventory component of Norton Administrator worked flawlessly, finding the same database. The metering feature reported that the licensing system was not enabled, even though it was fully configured on the server. When un-installing on the server, it could not remember its directories. After specifying the directories, it logged off the NT Server. Upon logging in again, apparently Norton Administrator had corrupted Windows NT since Windows Explorer and Program Manager were no longer present. Instead, there was just a blank screen.

OpenView Professional was unable to monitor applications. HP documentation states there is support using HP's DMI client for HP Vectra, which has many extensions to the DMI standard. DMI clients for non-HP machines were not tested. OpenView Professional was unable to adjust any application's server parameters.

To measure network utilization, OpenView Professional comes standard with HP's NetMetrix plug-in. This application crashed several times while we attempted to use its various functions. To make matters worse, NetMetrix also corrupted OpenView P rofessional, forcing a restart. Documentation mentions it works best with HP's own RMON probes. Although it can be configured with third-party probes, HP doesn't give a guarantee. OpenView Professional continued to be unstable with NetMetrix installed.

After un-installing this plug-in the software calmed down, but this meant OpenView Professional was unable to measure traffic utilization or throughput. It did allow the viewing of SNMP data related to IP and UDP traffic from a Cisco 7000 router, however. OpenView Professional did not support proactive management, security management or recognize unauthorized access.

When it comes to problems, OpenView Professional does provide alarm notification. The network manager can create traps and define custom alarm conditions. When an alarm occurs the Network Manager Console beeps, the network map displays an alarm condition and the event is logged into an alarm database. Often alerts went undetected over the background noise of the test lab. It would be nice to have an option for screen flashing or audible alerts outside of miscellaneous third-party add-ons.

One of our chief gripes with OpenView Professional is that it doesn't save session information between uses. Time and frustration could be saved if OpenView Professional automatically opened the last map each time it started. Many of the list boxes do not allow multiple selection. Another frustrating limitation was that the device access list was limited to each local map. This prevents any way to specify global defaults.

The map drawing tools are somewhat primitive as well. For example, there was no way to

change the symbol of a mapped device without first deleting it and then adding a new symbol. Also, the selection of symbols was limited. Furthermore, mapping tools did not support sub-net designations, or the topology definition. There was no way to rearrange or center maps on the display. On a map there was no way to distinguish between similar devices--all routers looked the sam e, all PCs look the same, etc.

There were no symbol selections to differentiate between different vendors or models for routers or servers other than by manually entering a name annotation. To distinguish between similar devices, the network manager must open the Discovery Manager Dialog and scroll through all the IP addresses for a description. Finally, no report capability was found in OpenView Professional or in any of the bundled add-ons. This is a serious limitation.

For the purpose of our test network, OpenView Professional was not a very flexible tool. On the upside, HP states there are more than 100 snap-in applications developed by third parties for OpenView Professional. This review did not evaluate any of these except the Norton Desklock Administrator. For an up-to-date list of all OpenView plug-in applications, check out www.hp.com/index.html, but remember that all these products may not support the new Professional version; be sure to check before buying.

Ope nView Professional pricing is based on two factors--the number of PC nodes and the number of servers managed. The starting configuration is a license for 10 users and one server and is priced at $1,495. Each additional server license is $695 or $5,950 for 10 servers. To manage additional PC nodes, HP refers the buyer to its VAR or reseller for McAfee or Symantec software. The McAfee Saber LAN Workstation is priced at $46 per node. The Symantec Norton Desklock Administrator for Windows is priced at $95 per node. A license fee for a typical 100-node IP network with five NetWare or Windows NT servers would be about $14,500. For a typical 250-node network, the HP Web site documentation recommends OpenView Network Node Manager rather than OpenView Professional.

Based on our test script and hands-on experience, this version of OpenView Professional is a tough fit for more than small-to-medium-sized LANs. While some problems were due to its beta status, we still had too much trouble with its base functionality a nd overall design.

In view of the scope of our test script, it's unfortunate that HP did not submit the more capable Node Manager product. As for the Professional version, we recommend it only for smaller IP and PC-centric networks.

Computer Associates Unicenter TNG

Unicenter TNG is designed to manage large enterprise networks, so it shouldn't come as a surprise that it's a big complex program. What you install depends on what you plan to manage from that particular Network Manager's Console. LANQuest Lab personnel installed Unicenter TNG without help from CA and had to reinstall a few times before it was done correctly. Most new Unicenter TNG customers would be advised to get CA help with installation.

For this test, Unicenter TNG was installed on a HP Vectra 500 P5-166 with 64 megabytes of RAM, a 1.5-gigabyte hard drive, NT 4.0 Server with Service Pack 3, and Microsoft SQL Server 6.5. All actions were done as the Administrator--no other user accounts were created on the machines. The first time the install program requested that SQL server be reinstalled with its sort order set to case-sensitive--this was not documented anywhere. After a couple of other mishaps, and with the install program set to wizard mode and typical components selected, installation of the server and Network Manager Console was completed. Installation of Unicenter TNG client agents was clear cut. TNG also includes an un-install icon to remove a client agent when needed. (There was only one other problem during testing. One client running the NT system agent for 24 hours crashed with a kernel page fault error. This crash corrupted the NT boot loader and NT Workstation had to be reinstalled at that node.)

When starting the 2D network map, running a discovery and opening the Enterprise Management console, memory usage reached 124 megabytes. The machine was too slow to use the 3D mapping. While the documentation recommends 64 megabytes of RAM

minimum, you should plan on a minimum of 128. And if 3D mapping is used, a fast graphics accelerator card is a must as well.

Even submitting to the 2D limitation, we were still a little disappointed. Once the TNG AutoDiscovery Wizard was launched, it only discovered nodes on two of the seven LAN segments. Even CA didn't know why, so we were forced to switch to the custom mode and explicitly define all our test sub-nets. Only then was TNG able to complete the discovery. Once a network segment was found, however, Unicenter TNG correctly identified all the connections to each router or switch, including all the interfaces on the routers. When discovery was completed, Unicenter had managed to find all our IP and IPX nodes; even multiple IP address machines were easily identified.

An optional Unicenter agent must be installed on each NT node to monitor systems information. TNG has built-in monitoring capabilities for Windows 95 and Unix devices, as well as SNMP-based hubs and routers. For this test TNG could monitor the 3Com and Cisco router s. It provided more information on the Cisco router than the 3Com.

TNG offers alternates to Map views that also are useful, such as a node listing called ObjectView. Similar to Windows Explorer with nested directories, ObjectView shows any additional information known about a specific node in a nested listing. To find added nodes to the network, discovery must be restarted. Then, after the nodes are added to the database, the map or topology browser must be manually refreshed. Nodes that are disconnected are reported, and alerts can be configured. The status change appears in the map display and in the alerts log, with the type of alert color-coded.

The standard Unicenter TNG can do some reconfiguring of selected routers and switches. As part of ObjectView, the configuration can be changed by re-entering the settings if the configurations are exposed in the management information base (MIB) for that device. The router and switch vendors used in your network may offer MIB-specific extensions f or plugging into TNG for additional device management.

Application management is only an option for TNG. There are many Unicenter components available from CA as well as from development partners. For a listing, refer to www.cai.com/products/uctr.htm.

The option for software distribution is Unicenter/Software Delivery. It provides centralized delivery of shrink-wrapped software. It also monitors concurrent license usage and provides audit reports for application usage. Unfortunately, this option was not provided.

Through the Agent Viewer in the standard TNG, process and services on a Unicenter client can be monitored but server parameters can't be changed. There is also an Optional Help Desk function for Unicenter TNG that allows centralized application help. The network software administrator can remotely view an application on another machine regardless of where the user is located, and then walk the user through using the application or solving an immediate problem--even going so far a s to take remote control of the keyboard.

In general, Unicenter TNG has good traffic monitoring features but limited characterization or performance management capabilities. The standard version monitors and displays the SNMP data it gathers from connected routers and switches. It also monitors remote RMON probes. There is an optional Unicenter Response Time Manager that collects network performance metrics. There are also third-party hardware and software RMON probes. For this test, Unicenter successfully monitored the LANQuest NetClarity software RMON probes on each network segment. Proactive performance measurement and traffic analysis add-ons are also available from Unicenter TNG Development Partners.

TNG can display router and switch utilization data that it collects from each connected device with SNMP. Alerts or alarms are generated when defined thresholds (IP, UDP, ICMP) are exceeded for each configured router or switch.

When alarms are generated, intelligent agents auto matically go to work to filter and correlate local problems to prevent escalation and isolate the problem. The alarm event is forwarded to the Unicenter Help Desk where a trouble ticket is opened and tracked until the problem is corrected.

TNG provides easy administration of user passwords with extensive features. User rights and access levels can be defined globally or on a server-by-server basis. User profiles can be built to monitor each user's network activity (this feature was not tested). Monitoring both authorized and unauthorized access is provided. Unauthorized access attempts are reported and can also trigger alarms.

Most of the time the Network Manager's Console will probably be left in the 2D map mode monitoring for alarms. Although the 3D mode is pretty slick in our tester's opinion, it's too time consuming. The 3D mode is not only a big strain on both the graphics card and the CPU, it's also a major RAM hog. Upgrading both the graphics card and your console workstation R AM will help, but we're not sure it's really worth it for network management on a daily basis.

TNG is visually appealing. It's easy to distinguish device types from their map symbols, and the list of available symbols is extensive.

Unicenter TNG has a Design Mode that lets the user customize the screen views before switching to the Run Mode. Unicenter is powerful and, no surprise, complex. Individual tools are well documented, but their relations to other tools within Unicenter need further clarification. For former Unix network managers, all of the tools can also be run from the command-line. From the 2D mode map the Network Manager can drill down to any node of interest whenever desired. If the node is running SNMP or an RMON probe, there's a lot of data. If you know what to look for navigating is easy and all the tools work seamlessly.

Unicenter TNG has three classes of reports: Network Summary, Objects and Links. These reports can be sent to a screen, printer or file. The Obj ects and Links have several sub-classes. The information is neatly arranged and pretty complete. Custom reports can also be defined and distributed. The only missing reports for this test were up-time or failure reports by network segment and by cluster of nodes.

As you would expect from an Enterprise class product, Unicenter TNG is very flexible and has the support of a large number of third-party products. All the major operating systems are supported. It automatically recognizes and monitors Cisco, 3Com and Bay Networks devices. It has built-in monitoring of RMON probes. For third-party developers, CA offers Software Development Kits to facilitate porting of applications to TNG. CA recently released Unicenter TNG Framework to improve the management of servers and workstations, and several hardware vendors such as NCR, Unisys and HP have already signed up.

The starting price for Unicenter TNG is only $2,500, but that's without any agents or add-in options. To manage a 100-node netwo rk with agents on five servers, the list price including the first year maintenance is about $37,000.

To manage a 250-node network with agents on 10 servers, the list price including the first year's maintenance is about $62,000. This translates to $250 to $370 per node. Contact CA or their third-party development partners for other options. A 100- or even 250-node network isn't classified as an Enterprise Network, but CA has made it financially attractive to install TNG and experience the power of heavyweight network management tools on a medium-sized network.

Yes, Unicenter TNG is powerful. Yes, there is a steep learning curve for its features. But it's worth it. After installation and network mapping is complete, and alert conditions set, a beginning network administrator can monitor the network and react properly to generated alarms. For the money, choose Unicenter TNG over the other products tested for a medium-sized heterogeneous network.

The return will be greater network manager efficiency and greater network up-time. For most organizations that translates into improved user productivity and better customer support.

What does it cost per user for your company to maintain your corporate data network each year? Is it $2,000, $5,000, $10,000--more? How many users and servers can one network manager support? Is it 200, 100, 50 or fewer? It's clear today that the only way to restrain cost-of-ownership for complex distributed enterprise networks is to invest in network management tools and training so managers can do their jobs more efficiently. The results

can be dramatic.

Some organizations have been successful in cutting cost-of-ownership by over 50 percent--enabling their network managers to double the number of users and servers supported.

Recent market studies by International Data Corp. and other analysts indicate that successfully implemented enterprise management systems can provide annual savings of $1,000 to $5,000 per user. If your company n etwork has 500 or more nodes and the cost to implement an enterprise management system is $75,000 to $100,000, that amounts to less than $200 per user. The cost savings make the decision a "no brainer." But, if your network is only 100 to 200 nodes, you might think twice before spending $500 per node.

For managers in this position, there are alternatives. Network management systems (NMS) for small-to-medium-sized networks can be implemented for under $20,000.

To help you make informed decisions about which network management platform best meets your needs, we set up a challenge match pitting two leading PC LAN-oriented network management systems, HP OpenView Professional and Novell's ManageWise, against Computer Associates International's Unicenter TNG, the steady leader of enterprise management system solutions. We wanted to find out how far we could take our middleweight contenders, and when it pays to bite the bullet and spring for the heavy guns.

This NMS challenge took place in July using a test network composed of routers, switches, multiple LAN segments, WAN connections, several network servers and over 60 workstations and PCs. Seven network management systems vendors, some enterprise heavyweights and some PC LAN middleweights were invited.

Of the heavyweights, only Computer Associates (CA) decided to climb into the ring. Though we asked Hewlett-Packard to participate on an enterprise level (something OpenView is capable of in certain versions), HP was only able to supply us with the smaller, Professional Suite version. Vendors who declined the challenge included IBM/Tivoli, Sun Microsystems, Cabletron and Platinum Technologies.

Overall, we were impressed by all the contenders, although in different ways. We not only reviewed each product as a whole, but also compared all three using specific criteria. For example, when it came to something as basic as installation, we found ManageWise to be straightforward and reliable, while the other two had some problems. Unicenter TNG is a big product with lots of installation options, meaning it took awhile to get everything properly configured. OpenView Professional should have had an easier install process as well, but in fairness we were working with a beta version of the new OpenView and notother products tested for a medium-sized heterogeneous network.

The return will be greater network manager efficiency and greater network up-time. For most organizations that translates into improved user productivity and better customer support.

```
Set     Items   Description
S1          0   S AU=(BURGMEIER H? OR BURGMEIER, H?)
S2          0   S HERMANN(2N)BURGMEIER

?  show files
```

[File 2] **INSPEC** 1898-2006/May W3

[File 6] **NTIS** 1964-2006/May W3

[File 8] **Ei Compendex(R)** 1970-2006/May W3

[File 34] **SciSearch(R) Cited Ref Sci** 1990-2006/May W3

[File 35] **Dissertation Abs Online** 1861-2006/May

[File 56] **Computer and Information Systems Abstracts** 1966-2006/May

[File 60] **ANTE: Abstracts in New Tech & Engineer** 1966-2006/May

[File 62] **SPIN(R)** 1975-2006/Mar W4

[File 65] **Inside Conferences** 1993-2006/May 31

[File 94] **JICST-EPlus** 1985-2006/Feb W4

[File 95] **TEME-Technology & Management** 1989-2006/May W4

[File 99] **Wilson Appl. Sci & Tech Abs** 1983-2006/Apr

[File 111] **TGG Natl.Newspaper Index(SM)** 1979-2006/May 22

[File 144] **Pascal** 1973-2006/May W1

[File 239] **Mathsci** 1940-2006/Jul

[File 256] **TecInfoSource** 82-2006/Jul

[File 434] **SciSearch(R) Cited Ref Sci** 1974-1989/Dec

?

```
Set      Items   Description
S1           0   S AU=(BURGMEIER H? OR BURGMEIER, H?)
S2           0   S HERMANN(2N)BURGMEIER
 ; show files
```

[File 9] **Business & Industry(R)** Jul/1994-2006/May 29
(c) 2006 The Gale Group. All rights reserved.

[File 13] **BAMP** 2006/May W3
(c) 2006 The Gale Group. All rights reserved.

[File 15] **ABI/Inform(R)** 1971-2006/May 31
(c) 2006 ProQuest Info&Learning. All rights reserved.

[File 16] **Gale Group PROMT(R)** 1990-2006/May 30
(c) 2006 The Gale Group. All rights reserved.

[File 20] **Dialog Global Reporter** 1997-2006/May 31
(c) 2006 Dialog. All rights reserved.

[File 47] **Gale Group Magazine DB(TM)** 1959-2006/May 30
(c) 2006 The Gale group. All rights reserved.

[File 75] **TGG Management Contents(R)** 86-2006/May W3
(c) 2006 The Gale Group. All rights reserved.

[File 88] **Gale Group Business A.R.T.S.** 1976-2006/May 23
(c) 2006 The Gale Group. All rights reserved.

[File 98] **General Sci Abs** 1984-2005/Jan
(c) 2006 The HW Wilson Co. All rights reserved.

[File 141] **Readers Guide** 1983-2006/Feb
(c) 2006 The HW Wilson Co. All rights reserved.

[File 148] **Gale Group Trade & Industry DB** 1976-2006/May 30
(c)2006 The Gale Group. All rights reserved.

[File 160] **Gale Group PROMT(R)** 1972-1989
(c) 1999 The Gale Group. All rights reserved.

[File 275] **Gale Group Computer DB(TM)** 1983-2006/May 29
(c) 2006 The Gale Group. All rights reserved.

[File 369] **New Scientist** 1994-2006/Apr W4
(c) 2006 Reed Business Information Ltd. All rights reserved.

[File 370] **Science** 1996-1999/Jul W3
(c) 1999 AAAS. All rights reserved.
**File 370: This file is closed (no updates). Use File 47 for more current information.*

[File 484] **Periodical Abs Plustext** 1986-2006/May W4

[File 553] **Wilson Bus. Abs.** 1982-2006/May

[File 610] **Business Wire** 1999-2006/May 31

**File 610: File 610 now contains data from 3/99 forward. Archive data (1986-2/99) is available in File 810.*

[File 613] **PR Newswire** 1999-2006/May 31

**File 613: File 613 now contains data from 5/99 forward. Archive data (1987-4/99) is available in File 813.*

[File 621] **Gale Group New Prod.Annou.(R)** 1985-2006/May 30

[File 624] **McGraw-Hill Publications** 1985-2006/May 31

**File 624: Homeland Security & Defense and 9 Platt energy journals added Please see HELP NEWS624 for more*

[File 634] **San Jose Mercury** Jun 1985-2006/May 28

[File 635] **Business Dateline(R)** 1985-2006/May 31

[File 636] **Gale Group Newsletter DB(TM)** 1987-2006/May 29

[File 647] **CMP Computer Fulltext** 1988-2006/Jun W3

[File 674] **Computer News Fulltext** 1989-2006/May W4

[File 696] **DIALOG Telecom. Newsletters** 1995-2006/May 30

[File 810] **Business Wire** 1986-1999/Feb 28

[File 813] **PR Newswire** 1987-1999/Apr 30

EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L1	8	burgmeier-h$.in.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT	OR	ON	2006/05/31 15:33

US 20040254945 A1 US-PGPUB
US 20040117392 A1 US-PGPUB
WO 2004102438 A2 EPO
WO 2004102438 A DERWENT
US 20040117392 A DERWENT

THIS APPLICATION

```
Set     Items   Description
S1     155759   S MAP? ? OR MAPP???
S2     417303   S GUI? ? OR (GRAPHIC? OR VISUAL?)(2W)(INTERFACE? OR APP? ? OR APPLICAT?)
OR DISPLAY? OR MENU? ? OR WEBPAGE?
S3      41159   S COMPUTER?()(SCREEN? OR WINDOW?) OR COMPUTER?(2N)(IMPLEMENT? OR TOOL? ?
OR UTILIT?)
S4      17879   S USERINTERFAC? OR (DROP OR PULL)()DOWN? OR (WINDOW OR DESKTOP)()MANAGER?
OR PULLDOWN?
S5      73664   S UI OR UIS OR (GRAPHIC? OR VISUAL? OR INFORMATION? OR DATA?)(2N)(MENU? OR
PAGE? OR SCREEN?)
S6     913497   S SOFTWARE()OBJECT? OR DATABASE? OR DOMAIN? OR COLUMN? OR TABLE? OR
DATAOBJECT? OR DATASTRUCTUR? OR DATAVALUE?
S7     986563   S OBJECT? OR DATAOBJECT? OR DATA()(ENTIT? OR STRUCTUR?) OR OBJECTMODEL? OR
OBJECT()MODEL?
S8      18060   S DATACOMMAND? OR DATAORDER? OR DATAUNIT? OR DATA()UNIT? OR DATA()MESSAG?
OR DATA()INSTRUCT?
S9      72924   S DATABYTE? OR DATABLOCK? OR DATA()(BLOCK? OR REQUEST? OR INPUT? OR TYPE?
OR MEMBER? OR VALUE?) OR DATATYPE?
S10    123020   S (DATA? OR SOFTWARE? OR OBJECT?)(2N)(IDENTIFIER? OR TAG? ? OR METADATA?
OR ADDRESS? OR LOCATER? OR LABEL? OR NAME? OR EXTENSION? OR DESIGNATION?)
S11    178201   S DATA()(REPRESENTATION? OR DEPOSITOR? OR REPOSITOR? OR BANK? OR FILE? OR
COLLECTION?) OR ROW OR ROWS
S12   1225241   S LOG OR LOGS OR CATALOG? OR CHRONICL? OR RECORD? ? OR LIST??? OR FIELD?
OR ARRAY? OR ARCHIV?
S13     88854   S S1 AND S2:S5 AND S6:S12
S14     87137   S MODULAT? OR CONVERT? OR MODIF? OR ALTER? OR TRANSFORM?
S15     85618   S CHANGE? OR CHANGING? OR TRANSLAT? OR AMEND? OR EDIT? OR UPDAT?
S16     53492   S CONVERSION? OR MANIPULAT? OR CUSTOMIZ? OR PERSONALIZ? OR INDIVIDUALIZ?
S17     71664   S CUSTOMIS? OR ADAPT? OR RECONFIGUR? OR CONFIGUR? OR REVIS? OR REARRANG?
S18      1217   S PERSONALIS? OR (CUSTOM OR TAILOR)()(MAKE? OR MADE? OR MAKING)
S19     16259   S REORDER? OR RESHUFFL? OR RESET? OR REFORMAT? OR READDRESS?
S20     88819   S WITHOUT OR LESS OR "NOT" OR MISSING? OR NO OR UN OR NON OR LACKING OR
ABSENT OR FREE OR MINUS OR ELIMINAT? OR SKIP? OR OMIT?
S21      5674   S COULDN? OR WOULDN? OR SHOULDN? OR DOESN? OR CAN()T OR WON()T
S22      9992   S UNMODULAT? OR NONMODULAT? OR UNCONVER? OR NONCONVER? OR UNMODIF? OR
NONMODIF? OR NONALTER? OR UNALTER? OR NONTRANSFORM? OR UNTRANSFORM?
S23     17894   S UNCHANG? OR NONCHANG? OR IMMUTAB? OR NONTRANSLAT? OR UNTRANSLAT? OR
UNAMEND? OR NONAMEND? OR UNEDIT? OR NONEDIT? OR NONUPDAT? OR UNUPDAT?
S24        93   S UNMANIPULAT? OR NONMANIPULAT? OR UNCUSTOM? OR NONCUSTOM? OR UNPERSONALI?
OR NONPERSONALI? OR NONINDIVIDUALI? OR UNINDIVIDUALI?
S25       367   S UNADAPT? OR NONADAPT? OR UNRECONFIGUR? OR NONRECONFIG? OR NONCONFIGUR?
OR UNCONFIGUR? OR UNREVIS? OR NONREVIS? OR NONREARRANG? OR UNREARRANG?
S26         3   S MODULATIONLESS? OR MODULATIONFREE? OR CHANGEFREE? OR CHANGELESS? OR
MODIFICATIONLESS? OR MODIFICATIONFREE?
S27         0   S AMENDMENTLESS? OR AMENDMENTFREE? OR EDITFREE? OR EDITLESS? OR
UPDATELESS? OR UPDATEFREE? OR REVISIONLESS? OR REVISIONFREE?
S28     62620   S SOURCE? OR ISSUER? OR ISSUING? OR SEMINAL?
S29     73936   S NUMBER()(ONE OR 1) OR PRINCIPAL? OR LEAD OR CONTROLLER? OR HEAD OR
MASTER OR BASIC?
S30     87803   S FIRST? OR 1ST OR PRIMARY OR INITIAL? OR ORIGINAL? OR LEADOFF? OR MAIN OR
CHIEF OR INTRODUCTORY?
S31     82368   S TARGET? OR RECEIV? OR RECEPT? OR RECIPIENT? OR DESTINATION? OR ENDPOINT?
S32     87923   S SECOND? OR 2ND OR ANOTHER OR AUXILIAR? OR BACKUP? OR EXTRA OR SLAVE? OR
SUPPLEMENT?
S33     86769   S SUBSIDIAR? OR DIFFERENT? OR ALTERNAT? OR NUMBER()(TWO OR 2)
S34     21856   S IC=G06F?
S35      6723   S S13 AND S1(10N)S2:S5(10N)S6:S12
S36      5972   S S35 AND S1:S12(10N)S14:S19(5N)S6:S12
S37       847   S S35 AND S1(10N)(S20:S21(5N)S14:S19 OR S22:S27)
S38       808   S S36 AND S37
S39       847   S S37:S38
```

```
S40           0   S S39 AND S1:S5(7N)(28:S30 AND S31:S33)(10N)S6:S12
S41         350   S S39 AND S34
S42         847   S S39 OR S41
S43         303   S S42 AND S1(5N)S2:S5(5N)S6:S12 AND S14:S19(5N)S6:S12 AND
S1(5N)(S20:S21(3N)S14:S19 OR S22:S27)
S44         197   S S43 AND AC=US/PR
S45         165   S S44 AND AY=(1970:2002)/PR
S46         163   S S44 NOT AY=(2003:2006)/PR
S47         165   S S45:S46
S48         106   S S43 NOT S44
S49          62   S S48 AND PY=1970:2002
S50          46   S S48 NOT PY=2003:2006
S51          84   S S48 AND AY=1970:2002
S52          78   S S48 NOT AY=2003:2006
S53          84   S S49:S52
S54         249   S S47 OR S53
S55         233   S S54 AND S1(5N)(S20:S21(3N)S14:S19)
S56          42   S S54 AND S1(5N)S22:S27
S57         249   S S55:S56
S58         249   IDPAT (sorted in duplicate/non-duplicate order)
 ; show files
```

[File 348] **EUROPEAN PATENTS** 1978-2006/ 200623
(c) 2006 European Patent Office. All rights reserved.
**File 348: For important information about IPCR/8 and forthcoming changes to the IC= index, see HELP NEWSIPCR.*

[File 349] **PCT FULLTEXT** 1979-2006/UB=20060608,UT=20060601
(c) 2006 WIPO/Univentio. All rights reserved.
**File 349: For important information about IPCR/8 and forthcoming changes to the IC= index, see HELP NEWSIPCR.*

58/3,K/53 (Item 53 from file: 349) **Links**
PCT FULLTEXT

00549731 **Image available**
METHODS AND DEVICES FOR MAPPING DATA FILES
PROCEDES ET DISPOSITIFS POUR CARTOGRAPHIER DES FICHIERS DE DONNEES
Patent Applicant/Assignee:
SYMTEC LIMITED,
CLIFTON-BLIGH Gervase,
Inventor(s):
CLIFTON-BLIGH Gervase,
Patent and Priority Information (Country, Number, Date):
Patent: WO 200013104 A1 20000309 (WO 0013104)
Application: WO 99GB2820 19990826 (PCT/WO GB9902820)
Priority Application: GB 9818633 19980826; GB 9824779 19981111; WO 98GB3481 19981120
Designated States:
(Protection type is "patent" unless otherwise stated - for applications prior to 2004)
AE AL AM AT AU AZ BA BB BG BR BY CA CH CN CR CU CZ DE DK DM EE ES FI GB
GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MD
MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT UA UG US
UZ VN YU ZA ZW GH GM KE LS MW SD SL SZ UG ZW AM AZ BY KG KZ MD RU TJ TM
AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM
GA GN GW ML MR NE SN TD TG
Publication Language: English
Fulltext Word Count: 30951

Fulltext Availability:
Detailed Description
Claims
Publication Year: 2000

Detailed Description
... representing
that i-th file; and
upon an user indicating a said j-th data file,
generating for each of one or more (j+1)-th data files
which are logically...

...layer is one layer beyond the
deepest layer which had previously been represented on
the map. Alternatively, the (j+1)-th files may be which
had not previously been displayed according to any of the
criteria discussed above (e.g. because the number of
(j...

...a predetermined value). The
additional region(s) may be drawn as a continuation of
the map (e.g. with the decreasing distance scales
continued for one more level) (this possibility is...

...the

See related box beneath

(j+1)-th files.

A twelth aspect of the invention is a device including display means such as a screen, and arranged to perform a method according to any of...

...may be part of, give access to or facilitate the management of any other data collection or network which may for example be based on the functioning of a physical hardware...aids), or a component of task specific software (e.g.

networked software, e-commerce software, menu or tool bars, data analysis tools) or hardware (e.g. for chip/circuit board layout...

...the usefulness of any of these facilities is severely limited by the size of the display device of a mobile telephone, which itself is limited by engineering constraints and present technology. By providing a mobile telephone with a display generated according to any of the methods of the invention defined above, the technical usefulness...

...significantly enhanced.

Normally, the device will include a mechanism for the user to input data (instructions), for example a touch sensitive screen. Alternatively or additionally, it may include a mechanical data input device, such as a joystick or a plurality of keys. In this case, the arrangement...

...additional regions of the control pad, preferably corresponds to the layout of the mechanical data input device.

For example, a mobile telephone may have buttons arranged in a keypad formation, and...

...the keys.

For example, if there are nine keys in the keypad in a 3x3 array (in addition to any further keys which may be present), for each i-th region...
...easy for a user to select one of the regions (files) using a mechanical data input device such as an (octagonal) joystick or button mouse which is not (or not necessarily...

...be related to the value M of the maximum number of (i+1)-th regions displayed. If there are more than M (i+1)-th files logically related to the i-th file, the extra regions may

be displayed separately (in a different area), or omitted (optionally symbol or symbols may be generated to...

...twelth aspects of the invention. The computer program product may be a computer program for implementing one of the methods of the invention stored on a recording medium, such as an...

...or one with a low resolution in relation to the number of files to be mapped (e.g. a total of no more than 10, 25 or 100 pixels per file...

...aspects above, the method may be able to reject (i.e. not include in the map) files according to their type. For example, especially in the context of the web, the map may consistently ignore audio and/or picture files, so that a large click distance can be displayed without over-complicating the map, or may create an additional region to representing files (or containing files) of a certain...

...reference to the accompanying figures in which.

Claim

1 A method of mapping hyperlinks between a plurality of data files comprising:
a step of, for a first file, in a display area displaying a first region and one or more second regions, said second regions respectively representing one...

...n-1) further steps, i = 21 ... n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one...

...th distance scale smaller than the (i-1)-th distance scale.

2 A method of mapping logical relations between a plurality of data files comprising:
a step of, for a first file, in a display area displaying a first region and one or more second regions, said second regions respectively representing one...

...n-1) further steps, i = 21 ... n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one...

...the resolution of the user's screen, (ii) the computer resources available for creating the display, (iii) the number of files a certain number of clicks away from the first file...

...user.

5 A method of moving between data files comprising the steps of:
generating a display of the interconnections between the data files by:
a step of, for a first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more...

...n-1) further steps, i = 21 ... n, of, for the or each i-th file, displaying one or more (i+1)- th regions, which respectively represent one or more (i+1...

...that the (i 1)-th distance scale;
selecting a file on the basis of the display; and
moving to the selected file.

6 A method according to claim 5 in which...

...structure and to substantially any leaf node.

7 A method of moving to a data file, said data file being one of a plurality of data files related by logical relations, the method comprising:
a step of, for a first file, in a display area defining a first region and one or more second regions, said second regions respectively...

...i
1)-th distance scale;
indicating (e.g. by a mouse) a position on said display area within one of said defined regions; and moving to the file represented by the region containing said position.

8 A method of mapping logical relations between a plurality of data files comprising:
a step of, for a first file, in a display area displaying a first region and one or more second regions, said second regions respectively representing one...

...n-1) further
steps, i = 21 ... n, of, for the or each i-th file, displaying one or more U+1)-th regions, said (i+1)-th regions respectively representing one...i and/or the number of said (i+1)-th
regions.

9 A method of mapping logical relations between a plurality of data files comprising:
a step of, for a first file, in a display area displaying a first region and, one or more second regions, said second regions respectively representing one...

...n-1) further steps, i = 21 ... n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one...

...determined by said
rule;
said rule being in relation to a predefined direction in the display area, whereby for i=1,..n the relative angular locations of the (i+i)-th regions are determined in relation to the predetermined direction.

10 A method of mapping logical relationships between a plurality of data files, comprising:
a step of defining a rule...

...other one of said data files;
a step of, for a first file, in a display area displaying a first region representing that i-th file and one or more second regions, said...

...n-1) further steps, i = 21 ... n, of, for the or
each i-th file, displaying one or more (i+1)-th regions,
said (i+1)-th regions respectively representing one...

...th file,
and being spatially related to the i-th region.

11 A method of displaying interconnections between a plurality of data files comprising, upon a user's location changing to correspond to a first file:
a step of, for the first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more...

...n-1) further steps, i = 21 ... n, of, for the or
each i-th file, displaying one or more (i+1)-th regions,
which respectively represent one or more (i+1...

...th region representing
that i-th files.

12 A method of moving to a data file comprising the steps of:
generating a display of the interconnections between a plurality of data files by:
a step of, for a first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more...

...n-1) further steps, i = 21 ... n, of, for the or
each i-th file, displaying one or more (i+1)-th regions,
which respectively represent one or more (i+1...

...the i-th region representing
that i-th file;
for a selected (j)-th data file (1<j<n) represented
by one of said regions generating for each of said one...

...and
upon a user indicating a said respective additional region, moving to the corresponding data file.

13 A method of mapping a plurality of data files comprising the steps of:
generating a display of the interconnections between the plurality of data files by:
a step of, for a first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more...

...n-1) further steps, i = 2, ... n, of, for the or
each i-th file, displaying one or more (i+1)-th regions,
which respectively represent one or more (i+1...

...file; and
upon said user controlling an indicator device so that a position within the map determined by the physical arrangement of said indicator device is within a said region representing a j-th data file generating for each of one or more related data files which have a predetermined logical...

...of moving to a file, comprising generating by a method according to claim 13 a map of a plurality of data files having said file as one of said (j+1...

...mouse, and said additional control signal includes a mouse click motion.

16 A method of displaying interconnections between a plurality of data files comprising the steps of: generating a display of the interconnections between the data files by:
a step of, for a first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more...

...n-1) further steps, i = 21 ... n, of, for the or each i-th file, displaying one or more (i+1)-th regions, which respectively represent one or more (i+1...

...representing that i-th file; and
upon an user indicating a said j-th data file, generating for each of one or more (j+1)-th data files which are logically...

...represented, a respective additional region representing the (j+1)-th file.

17 A method of mapping logical relations between a plurality of data files comprising:
a step of, for a first file, in a display area displaying a first region and, one or more second regions, said second regions respectively representing one...

...n-1) further steps, i = 21 ... n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one...

...to each other with an angular relationship determined by said rule.

18 A method of mapping logical relations between a plurality of data files comprising:
a step of, for a first file, in a display area displaying a first region and, for M an integer greater

than one and q an integer...

...the or
each i-th file, for qj an integer in the range 1, ... IM displaying qj (i+1)-th regions, said (i+1)-th regions respectively representing one or more...

...18 in which said
predetermined positions are in relation to a predetermined direction in said display.

20 A method according to any of claims 9 to 19 wherein said second regions are displayed according to a first distance scale, and for each i-th file said (i+1)-th regions are displayed according to a respective i-th distance scale smaller than the respective (i-1)-th...

...predetermined relation
such that irrespective of the value of n the total area of the display never exceeds a predetermined value.

24 A method according to any of claims 1 to...
...claim 31 in which said rule
is in relation to a predefined direction in the display area, whereby the angular relationships of the (i+1)-th regions are determined in relation...A method according to any preceding claim in which
the first file is a data file which corresponds to a present browser location of a user.

36 A method according to...

...of
(i+1)-th regions logically related to the i-th file, the method not displaying regions for said subset of identified (i+1)-th files.

38 A method according to...

...at
least one said criterion is that an identified file is already represented on the map, and optionally also that the data file to which it is logically related is already represented on the map.

39 A method according to claim 38 in which said at least one criterion is...

...i+1)-th data files logically
related to a given i-th file, the method displays M (i+1)-th regions, respectively representing only M of those (i+1)-th files.

40 A method of mapping logical relationships of a data

file, which includes a step of selecting a file, and causing the terminal to perform the method according to any of claims for a set of files including said data file, employing said data file as said first file.

41 A method according to any preceding claim wherein the user can label at least one said file, the map modifying the region which represents each of said respective labelled data files to indicate that the file has been labelled.

42 A method according to any...

...that region.

43 A method according to any preceding claim in which additional information is displayed to indicate the significance of at least one of said data files.

44 A method according to claim 43 in which said additional information is displayed upon a command by the user.

45 A method of mapping interconnections between plurality of segments of one or more data files, comprising:
a step of, for a first segment, in a display area displaying a first region and one or more second regions, which respectively represent one or more...

...steps, i = 21 ... n, of for the or each i-th portion of the file displaying one or more (i+1)-th regions, which respectively represent one or more (i+1...

...the shape of the regions and/or their size and/or their arrangement within the display.

53 A method according to any preceding claim further comprising a step of generating a second map, including areas respectively representing (i+1)-th regions of a given i-th region, said...

...i+1) -th regions.

54 A method according to claim 53 in which the second map is spatially separated in the display area from the map representing the plurality of data files.

55 A method according to any preceding claim in...

...which for n said (i+1)-th regions are

substantially square.

58 A device including display means, and arranged to perform a method according to any of the preceding claims.

59...

...A device according to any of claims 58 to 61 which includes a mechanical data input device, the arrangement of the regions corresponding to the layout of the mechanical data input device.

63 A computer program product which can be read by a data processing device...

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 : G06F 17/30 — A1

(11) International Publication Number: WO 00/13104

(43) International Publication Date: 9 March 2000 (09.03.00)

(21) International Application Number: PCT/GB99/02820

(22) International Filing Date: 26 August 1999 (26.08.99)

(30) Priority Data:

9818633.1	26 August 1998 (26.08.98)	GB
9824779.4	11 November 1998 (11.11.98)	GB
PCT/GB98/03481	20 November 1998 (20.11.98)	GB

(71) Applicant *(for all designated States except US)*: SYMTEC LIMITED [GB/GB]; 32 Athol Street, Douglas, Isle of Man IM1 1JB (GB).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: CLIFTON-BLIGH, Gervase [GB/GB]; 14C Pembridge Road, London W11 3HL (GB).

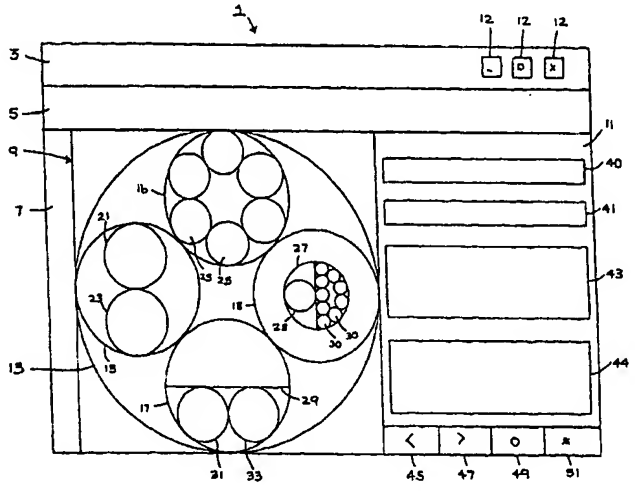
(74) Agents: WATKIN, Timothy, L. et al.; Lloyd Wise, Tregear & Co, Commonwealth House, 1-19 New Oxford Street, London WC1A 1LW (GB).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published

With international search report.

(54) Title: METHODS AND DEVICES FOR MAPPING DATA FILES



(57) Abstract

A method of mapping interconnections between a plurality of data files, such as files of the world wide web. The method comprises a step of, for a first file, in a display area displaying a first region. Then, according to a first distance scale, displaying one or more second regions, which respectively represent one or more second files logically related to the first file, for example by hyperlinks. The second regions are spatially related in a display to the first region. In (n-1) further steps, i = 2,..., n, of for the or each i-th file the method displays according to an i-th distance scale smaller than the (i-1) -th distance scale one or more (i+1) -th regions. Each of these (i+1) -th regions respectively represent one or more (i+1) -th files which are logically related to the i-th file, and they are spatially related on a display to the i-th region representing that i-th file. The method may be implemented in software, for example in a web browser.

Claims

1. A method of mapping hyperlinks between a plurality of data files comprising:

a step of, for a first file, in a display area displaying a first region and one or more second regions, said second regions respectively representing one or more second files which are accessible from the i-th file by a hyperlink, and being spatially related to the first region, the sizes of said second regions being according to a first distance scale; and

(n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one or more (i+1)-th files which are accessible from the i-th file by a hyperlink to the i-th file, and being spatially related to the i-th region representing that i-th file, the sizes of said second regions being according to an i-th distance scale smaller than the (i-1)-th distance scale.

2. A method of mapping logical relations between a plurality of data files comprising:

a step of, for a first file, in a display area displaying a first region and one or more second regions, said second regions respectively representing one or more second files logically related to the first file and being spatially related to the first region, the sizes of

said second regions being according to a first distance scale; and

for a predetermined value of n, (n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one or more (i+1)-th files which are logically related to the i-th file, and being spatially related to the i-th region representing that i-th file, the sizes of said (i+1)-th regions being according to an i-th distance scale smaller than the (i-1)-th distance scale.

3. A method according to claim 2 in which the value of n is at least 3.

4. A method according to claim 2 or claim 3 wherein n depends upon any one or more of: (i) the resolution of the user's screen, (ii) the computer resources available for creating the display, (iii) the number of files a certain number of clicks away from the first file, or (iv) an earlier selection by the user.

5. A method of moving between data files comprising the steps of:

generating a display of the interconnections between the data files by:

a step of, for a first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more second files logically related to the first file, and which are

spatially related to the first region, the sizes of said second regions being according to a first distance scale; and

(n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, which respectively represent one or more (i+1)-th files which are logically related to the i-th file, and which are spatially related to the i-th region representing that i-th file, the sizes of said (i+1)-th regions being according to an i-th distance scale smaller that the (i-1)-th distance scale;

selecting a file on the basis of the display; and

moving to the selected file.

6. A method according to claim 5 in which said data files are a directory structure, the user is able to move to substantially any directory of the directory structure and to substantially any leaf node.

7. A method of moving to a data file, said data file being one of a plurality of data files related by logical relations, the method comprising:

a step of, for a first file, in a display area defining a first region and one or more second regions, said second regions respectively representing one or more second files logically related to the first file and being spatially related to the first region, the sizes of said second regions being according to a first distance scale;

for a predetermined value of n, (n-1) further steps, i = 2,...,n, of, for the or each i-th file, defining one or more (i+1)-th regions, said (i+1)-th regions respectively representing one or more (i+1)-th files which are logically related to the i-th file, and being spatially related to the i-th region representing that i-th file, the sizes of said (i+1)-th regions being according to an i-th distance scale smaller than the (i-1)-th distance scale;

indicating (e.g. by a mouse) a position on said display area within one of said defined regions; and

moving to the file represented by the region containing said position.

8. A method of mapping logical relations between a plurality of data files comprising:

a step of, for a first file, in a display area displaying a first region and one or more second regions, said second regions respectively representing one or more second files logically related to the first file and being spatially related to the first region, the sizes of said second regions being according to a first distance scale, and the shape of said second regions being determined by a rule; and

for a predetermined value of n, (n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one or more (i+1)-th

files which are logically related to the i-th file, and being spatially related to the i-th region representing that i-th file, the sizes of said (i+1)-th regions being according to an i-th distance scale smaller than the (i-1)-th distance scale, and the shapes of said (i+1)-th regions being determined by said rule, said rule being independent of i and/or the number of said (i+1)-th regions.

9. A method of mapping logical relations between a plurality of data files comprising:

a step of, for a first file, in a display area displaying a first region and, one or more second regions, said second regions respectively representing one or more second files logically related to the first file, being spatially arranged in relation to the first region, and being spatially arranged relative to each other with an angular relationship determined by a rule, and

(n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one or more (i+1)-th files which are logically related to the i-th file, being spatially arranged in relation to the i-th region, and being spatially arranged relative to each other with an angular relationship determined by said rule;

said rule being in relation to a predefined

direction in the display area, whereby for i=1,..n the relative angular locations of the (i+1)-th regions are determined in relation to the predetermined direction.

10. A method of mapping logical relationships between a plurality of data files, comprising:

a step of defining a rule which determines whether any one of said data files is logically related to any other one of said data files;

a step of, for a first file, in a display area displaying a first region representing that i-th file and one or more second regions, said second files respectively representing one or more second files and being spatially related to the first region; and

(n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one or more (i+1)-th files logically related to the i-th file, and being spatially related to the i-th region.

11. A method of displaying interconnections between a plurality of data files comprising, upon a user's location changing to correspond to a first file:

a step of, for the first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more second files logically related to the first file, and which are spatially related to the first region; and

(n-1) further steps, i = 2,...,n, of, for the or

each i-th file, displaying one or more (i+1)-th regions, which respectively represent one or more (i+1)-th files which are logically related to the i-th file, and which are spatially related to the i-th region representing that i-th files.

12. A method of moving to a data file comprising the steps of:

generating a display of the interconnections between a plurality of data files by:

a step of, for a first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more second files logically related to the first file, and which are spatially related to the first region; and

(n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, which respectively represent one or more (i+1)-th files which are logically related to the i-th file, and which are spatially related to the i-th region representing that i-th file;

for a selected (j)-th data file ($1<j\leq n$) represented by one of said regions generating for each of said one or more (j+1)-th data files which are logically related to (j)-th file a respective additional region; and

upon a user indicating a said respective additional region, moving to the corresponding data file.

13. A method of mapping a plurality of data files

comprising the steps of:

generating a display of the interconnections between the plurality of data files by:

a step of, for a first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more second files logically related to the first file, and which are spatially related to the first region; and

(n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, which respectively represent one or more (i+1)-th files which are logically related to the i-th file, and which are spatially related to the i-th region representing that i-th file; and

upon said user controlling an indicator device so that a position within the map determined by the physical arrangement of said indicator device is within a said region representing a j-th data file generating for each of one or more related data files which have a predetermined logical connection to the (j)-th file a respective additional region including data indicating the significance of said respective connected file.

14. A method of moving to a file, comprising generating by a method according to claim 13 a map of a plurality of data files having said file as one of said (j+1)-th files, the method further comprising upon the user supplying an additional control signal when the position

determined by the physical arrangement of said indicator is a position corresponding to that (j+1)-th file, moving to that (j+1)-th file.

15. A method according to claim 13 or claim 14 in which said indicator device is a mouse, and said additional control signal includes a mouse click motion.

16. A method of displaying interconnections between a plurality of data files comprising the steps of:

generating a display of the interconnections between the data files by:

a step of, for a first file, in a display area displaying a first region and one or more second regions, which respectively represent one or more second files logically related to the first file, and which are spatially related to the first region; and

(n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, which respectively represent one or more (i+1)-th files which are logically related to the i-th file, and which are spatially related to the i-th region representing that i-th file; and

upon an user indicating a said j-th data file, generating for each of one or more (j+1)-th data files which are logically related to said j-th file and which were not already represented, a respective additional region representing the (j+1)-th file.

17. A method of mapping logical relations between a

plurality of data files comprising:

a step of, for a first file, in a display area displaying a first region and, one or more second regions, said second regions respectively representing one or more second files logically related to the first file, being spatially arranged in a two-dimensional formation in relation to the first region, and being spatially arranged relative to each other with an angular relationship determined by a rule, and

(n-1) further steps, i = 2,...,n, of, for the or each i-th file, displaying one or more (i+1)-th regions, said (i+1)-th regions respectively representing one or more (i+1)-th files which are logically related to the i-th file, being spatially arranged in a two-dimensional formation in relation to the i-th region, and being spatially arranged relative to each other with an angular relationship determined by said rule.

18. A method of mapping logical relations between a plurality of data files comprising:

a step of, for a first file, in a display area displaying a first region and, for M an integer greater than one and q an integer in the range 1,..M, q second regions, said second regions respectively representing one or more second files logically related to the first file, being arranged in relation to the first region, having a relative angular relationship according to the first q positions of a predefined sequence of M

predetermined positions.

(n-1) further steps, i = 2,...,n, of, for the or each i-th file, for q_i an integer in the range 1,...,M displaying q_i (i+1)-th regions, said (i+1)-th regions respectively representing one or more (i+1)-th files which are logically related to the i-th file, being arranged in relation to the i-rh region, and having a relative angular relationship according to the first q_i positions of said sequence of predetermined positions.

19 A method according to claim 18 in which said predetermined positions are in relation to a predetermined direction in said display.

20. A method according to any of claims 9 to 19 wherein said second regions are displayed according to a first distance scale, and for each i-th file said (i+1)-th regions are displayed according to a respective i-th distance scale smaller than the respective (i-1)-th distance scale.

21. A method according to claim 20 wherein said second regions have a size according to said first distance scale and said (i+1)-th regions have a size according to said i-th distance scale.

22. A method according to any of claims 1 to 8, or 21, in which the respective sizes of the (i+1)-th regions of a given i-th file are the i-th distance scale multiplied by value which is a function of a variable characterizing the respective (i+1)-th file.

23. A method according to any of claims 1 to 8, or 20 to 22, in which said distance scales are chosen to decrease with i according to a predetermined relation such that irrespective of the value of n the total area of the display never exceeds a predetermined value.

24. A method according to any of claims 1 to 8, or 20 to 23 in which the distance scales are chosen such that the total area of the one or more (i+1)-th regions for each i-th file is less than the area of the i-th region representing the i-th file.

25. A method according to any preceding claim in which the data files define a tree-like (preferably tree) directory structure and the logical relationships between the data files are the links of the directory structure.

26. A method according to any preceding claim in which said logical relations are defined by hyperlinks between said data files.

27. A method according to claim 1 or claim 26 in which said hyperlinks are hypertext links.

28. A method according to claim 26 or 27 in which the logical relationships are such that each said (i+1)-th file can be reached from the i-th file by one click.

29. A method according to claim 1, or claims 26 to 28 in which said data files are files of the world wide web.

30. A method according to any of claims 1 or 26 to 29 in which the plurality of files are provided by a plurality of spatially separated servers.

31. A method according to any preceding claim in which said second regions are spatially arranged relative to each other with an angular relationship determined by a rule, and for i = 2,...,n, the or each said (i+1)-th regions are spatially arranged relative to each other with an angular relationship determined by said rule.

32. A method according to claim 31 in which said rule is in relation to a predefined direction in the display area, whereby the angular relationships of the (i+1)-th regions are determined in relation to the predetermined direction.

33. A method according to any preceding claim wherein if a given one of said files is a frame set, the region which represents the file indicates this.

34. A method according to claim 33 in which the region representing an i-th file which is a frame set partitioned into l sections (k=1,...l), the respective i-th region which represents the i-th file is partitioned into l sections (k=1,...,l), the (i+1)-th regions which represent those files which are linked to the i-th file by anchors within the k-th section of the i-th file being within the k-th section of the i-th region.

35. A method according to any preceding claim in which the first file is a data file which corresponds to a present browser location of a user.

36. A method according to claim 35 which is performed upon the present web or system browser location of the

user changing.

37, A method according to any preceding claim, wherein at least one criterion is used to identify a subset of (i+1)-th regions logically related to the i-th file, the method not displaying regions for said subset of identified (i+1)-th files.

38. A method according to claim 37 in which said at least one said criterion is that an identified file is already represented on the map, and optionally also that the data file to which it is logically related is already represented on the map.

39. A method according to claim 38 in which said at least one criterion is such that if there are more than a predetermined number M of (i+1)-th data files logically related to a given i-th file, the method displays M (i+1)-th regions, respectively representing only M of those (i+1)-th files.

40. A method of mapping logical relationships of a data file, which includes a step of selecting a file, and causing the terminal to perform the method according to any of claims for a set of files including said data file, employing said data file as said first file.

41. A method according to any preceding claim wherein the user can label at least one said file, the map modifying the region which represents each of said respective labelled data files to indicate that the file has been labelled.

42. A method according to any preceding claim wherein the shape or colouring of at least one said region is in accordance with characteristics of the respective file represented by that region.

43. A method according to any preceding claim in which additional information is displayed to indicate the significance of at least one of said data files.

44. A method according to claim 43 in which said additional information is displayed upon a command by the user.

45. A method of mapping interconnections between plurality of segments of one or more data files, comprising:

a step of, for a first segment, in a display area displaying a first region and one or more second regions, which respectively represent one or more second segments logically related to the first segment, and which are spatially related to the first region; and

(n-1) further steps, i = 2,...,n, of for the or each i-th portion of the file displaying one or more (i+1)-th regions, which respectively represent one or more (i+1)-th segments which are logically related to the i-th segment, and which are spatially related to the i-th region representing that i-th segment.

46. A method according to claim 45 wherein said segments of data files represent respective portions of a structured computer language, the logical relationships

corresponding to said structure.

47. A method according to claim 46 wherein the computer language is HMTL.

48. A method according to any preceding claim in which for i=1,...n the (i+1)-th regions are arranged in a two dimensional formation.

49. A method according to any preceding claim in which the (i+1)-th regions for each i-th region are non-overlapping within the i-th region.

50. A method according to any preceding claim wherein the (i+1)-th regions are as large as possible within the constraint of a predetermined size and arrangement rule.

51. A method according to any preceding claim wherein the (i+1)-th regions are arranged symmetrically within the corresponding i-th region.

52. A method according to any preceding claim wherein the user has the option of redefining the shape of the regions and/or their size and/or their arrangement within the display.

53. A method according to any preceding claim further comprising a step of generating a second map, including areas respectively representing (i+1)-th regions of a given i-th region, said areas having relative positions corresponding to the relative positions of the (i+1) -th regions.

54. A method according to claim 53 in which the second map is spatially separated in the display area from the

map representing the plurality of data files.

55. A method according to any preceding claim in which for i=1,.., n said (i+1)-th regions are substantially circular.

56. A method according to any of claims 1 to 54 in which for i=1,.., n said (i+1)-th regions are substantially rectangular.

57. A method according to any of claims 1 to 54 in which for i=1,.., n said (i+1)-th regions are substantially square.

58. A device including display means, and arranged to perform a method according to any of the preceding claims.

59. A device according to claim 58 which is a portable device.

60. A device according to claim 58 which is a mobile telephone.

61. A device according to claim 58 which is a fixed installation.

62. A device according to any of claims 58 to 61 which includes a mechanical data input device, the arrangement of the regions corresponding to the layout of the mechanical data input device.

63. A computer program product which can be read by a data processing device to cause the data processing device to perform a method according to any of claims 1 to 57.

58/3,K/101 (Item 101 from file: 348) **Links**
EUROPEAN PATENTS

00833416
Filtering system and method for a high performance network management map
System und Verfahren zur Filterung eines Hochleistungsnetzwerk-Verwaltungsplans
Systeme et methode de filtrage pour un plan de gestion d'un reseau a haute performance
PATENT ASSIGNEE:
Hewlett-Packard Company, (3016020), 3000 Hanover Street, Palo Alto, CA 94304, (US), (Proprietor designated states: all)
INVENTOR:
Schettler, Robert Dwight, 1027 Summer Street, Fort Collins, CO 80521, (US)
Pulsipher, Eric A., 2937 Redburn Dr., Fort Collins, CO 80525, (US)
Atkins, Brian J., 2918 Eindborough Dr., Fort Collins, CO 80525, (US)
LEGAL REPRESENTATIVE:
Schoppe, Fritz (55463), Schoppe, Zimmermann, Stockeler & Zinkler Patentanwalte Postfach 246, 82043 Pullach bei Munchen, (DE)
PATENT (CC, No, Kind, Date): EP 772318 A2 970507 (Basic)
EP 772318 A3 990113
EP 772318 B1 051228
APPLICATION (CC, No, Date): EP 96116263 961010;
PRIORITY (CC, No, Date): US 551499 951101
DESIGNATED STATES: DE; FR; GB; SE
INTERNATIONAL PATENT CLASS (V7): H04L-012/24
ABSTRACT WORD COUNT: 129
NOTE:
Figure number on first page: 1

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPAB97	982
CLAIMS B	(English)	200552	659
CLAIMS B	(German)	200552	728
CLAIMS B	(French)	200552	759
SPEC A	(English)	EPAB97	10683
SPEC B	(English)	200552	10857
Total word count - document A			11666
Total word count - document B			13003
Total word count - documents A + B			24669

see US equivalent - attached

SPECIFICATION \FIELD
FIELD OF THE INVENTION
The present invention generally relates to data communication networks and, more particularly...
...on-demand submap systems, a submap corresponds with each view of the network to be displayed. The network management map is the collection of all submaps. In these on-demand submap systems, and particularly the...

...user wishes to have available, and hence, specifies the submaps which are resident within the map. Moreover, the user can also open, or "explode," a submap during operation even though it is not specified as resident in the map. In this case, the submap is generated immediately from the topology data when the user...

...network topology is discovered, and all this topology information is displayed in the network management map. This predicament results in clutter of objects in the submaps and in dilution of the functionality pertaining to each submap. Furthermore, this...

...contents of the network management map for a management station for the purpose of reducing object clutter, minimizing memory requirements, and minimizing expense, and optimizing performance (including speed).

SUMMARY OF THE...

...system and method for a management station for customizing the contents of a network management map. The system comprises a processor which executes the instructions provided by the various software elements of the system, a memory for storing the various software elements, a display for showing the devices and interconnections of the network, an interface that interconnects the foregoing...

...determining the network topology data, a layout mechanism for converting the network topology data to map data and for driving the display with the map data, and a filtering system, which is a significant feature of the present invention as...

...situated between the discovery mechanism and the layout mechanism so that the filtering system filters objects within the topology data that pass from the discovery mechanism to the layout mechanism. Second...

...be situated between the layout mechanism and the network so that the filtering system filters objects within the topology data that pass from the network to the discovery mechanism. Third, the filtering system can also be situated between discovery mechanisms so that the filtering system filters objects within the topology data that passes between the discovery mechanisms.

In an implementation where more...

...filtering systems utilize a common filtering library, which contains the filtering specification pertaining to the objects. The filtering specification associated with each filtering system can include a list of one or more objects to be allowed or disallowed, a boolean expression (or equation) defining which objects are to be allowed or disallowed, or any other mechanism for specifying a filtering condition...

...the filtering system and method is that they customize the contents of a network management map generated by a management station so as to reduce clutter of objects in submaps.

Another advantage of the filtering system and method is that they minimize memory requirements as well as resultant expense for generating a network management map in a management station.

Another advantage of the filtering system and method is that they enhance the performance of a process for generating a network management map in a management station.

Another advantage of the filtering system and method is that they minimize requisite processing time for producing a network management map in a management station.

Another advantage of the filtering system and method is that they minimize requisite interprocess communication in a management station for generating a network management map of a data communications network.

Another advantage of the filtering system and method is that...

...in the art upon examination of the following drawings and detailed description. All such additional objects, features, and advantages are intended to be included herein within the scope of the present...

...method of the present invention;

CLAIMS 1. A management system (100) for efficiently discovering and displaying devices and interconnections of a network (118), comprising:

a processor (102);

a memory (110);

a display (108);

an interface (104) interconnecting said processor (102), said memory (110), and said display (108) and capable of connecting to said network (118);

a discovery mechanism (302) stored in...

...data (316) from said discovery mechanism (302), said layout mechanism (304) configured to drive said display (108) based upon said topology data (316); and

a filtering system (103) stored in said memory (110) for driving said processor (102), said filtering system (103) configured to filter objects within said topology data (316) that pass from said discovery mechanism (302) to said layout...

...stored in said memory (110) for driving said processor (102), said second filtering system (103) configured to filter objects within said topology data (316) that pass from said network (118) to said discovery mechanism...

...stored in said memory (110) for driving said processor (102), said second filtering system (103) configured to filter objects within said topology data (316) that pass between said first and second discovery mechanisms (302...

...stored in said memory (110) for driving said processor (102), said third filtering system (103) configured to filter objects within said topology data (316) that pass from said network (118) to said discovery mechanism...

...321) in communication with said first, second, and third filtering systems (103), said library (321) configured to specify which of said objects are communicated through said filtering systems (103).

6. A management system (100) for efficiently discovering and displaying devices and interconnections of a network (118), comprising:

a processor (102);

a memory (110);

a display (108);

an interface interconnecting said processor (102), said memory (110), and said display (108) and capable of connecting to said network (118);

a discovery mechanism (302) stored in...

...data (316) from said discovery mechanism (302), said layout mechanism (304) configured to drive said display (108) based upon said topology data (316); and

a filtering system (103) stored in said memory (110) for driving said processor (102), said filtering system (103) configured to filter objects within said topology data (316) that pass from said network (118) to said discovery mechanism (302).

7. A management system (100) for efficiently discovering and displaying

devices and interconnections of a network (118), comprising:
a processor (102);
a memory (110);
a display (108);
an interface (104) interconnecting said processor (102), said memory (110), and said display (108) and capable of connecting to said network (118);
first and second discovery mechanisms (302...

...data (316) from said discovery mechanism (302), said layout mechanism (304) configured to drive said display (108) based upon said topology data (316); and
a filtering system (103) stored in said...

...said topology data (316) with a predefined library (321) to determine allowable objects and nonallowable objects;
converting said allowable objects into map data (328) and displaying said map data (328); and
refraining from converting said nonallowable objects into map data (328) and from displaying said map data (328).

9. A filtering method (101) for discovering and displaying devices and interconnections of a network (118), comprising the steps of:
generating topology data (316...

...comparing objects within said topology data (316) with a predefined library (321) to determine allowable objects and nonallowable objects;
converting said allowable objects into map data (328) and displaying said map data (328); and
eliminating said nonallowable objects from said topology data (316).

10. A filtering method (101) for discovering and displaying devices and interconnections of a network (118), comprising the steps of:
generating topology data (316...

...with first and second discovery mechanisms (302) that are interfaced to said network (118);
communicating objects within said topology data (316) between said first and second discovery mechanisms (302);
comparing said objects that are communicated between said first and second discovery mechanisms (302) with a predefined library (321) to determine allowable objects and nonallowable objects; and
preventing transfer of said nonallowable objects between said first and second discovery mechanisms (302), while permitting transfer of said allowable objects therebetween.

US005787252A

United States Patent [19]

Schettler et al.

[11] Patent Number: 5,787,252

[45] Date of Patent: Jul. 28, 1998

[54] **FILTERING SYSTEM AND METHOD FOR HIGH PERFORMANCE NETWORK MANAGEMENT MAP**

[75] Inventors: **Robert Dwight Schettler**; **Eric A. Pulsipher**; **Brian J. Atkins**, all of Fort Collins, Colo.

[73] Assignee: **Hewlett-Packard Company**, Palo Alto, Calif.

[21] Appl. No.: **551,499**

[22] Filed: **Nov. 1, 1995**

[51] **Int. Cl.**[6] **G06F 15/177**

[52] **U.S. Cl.** **395/200.54**; 345/335; 395/284; 395/830; 395/200.1; 395/828; 711/170

[58] **Field of Search** 395/497.01, 284, 395/200.1, 653, 828, 830, 832, 834, 200.54; 345/333, 334, 335

[56] **References Cited**

U.S. PATENT DOCUMENTS

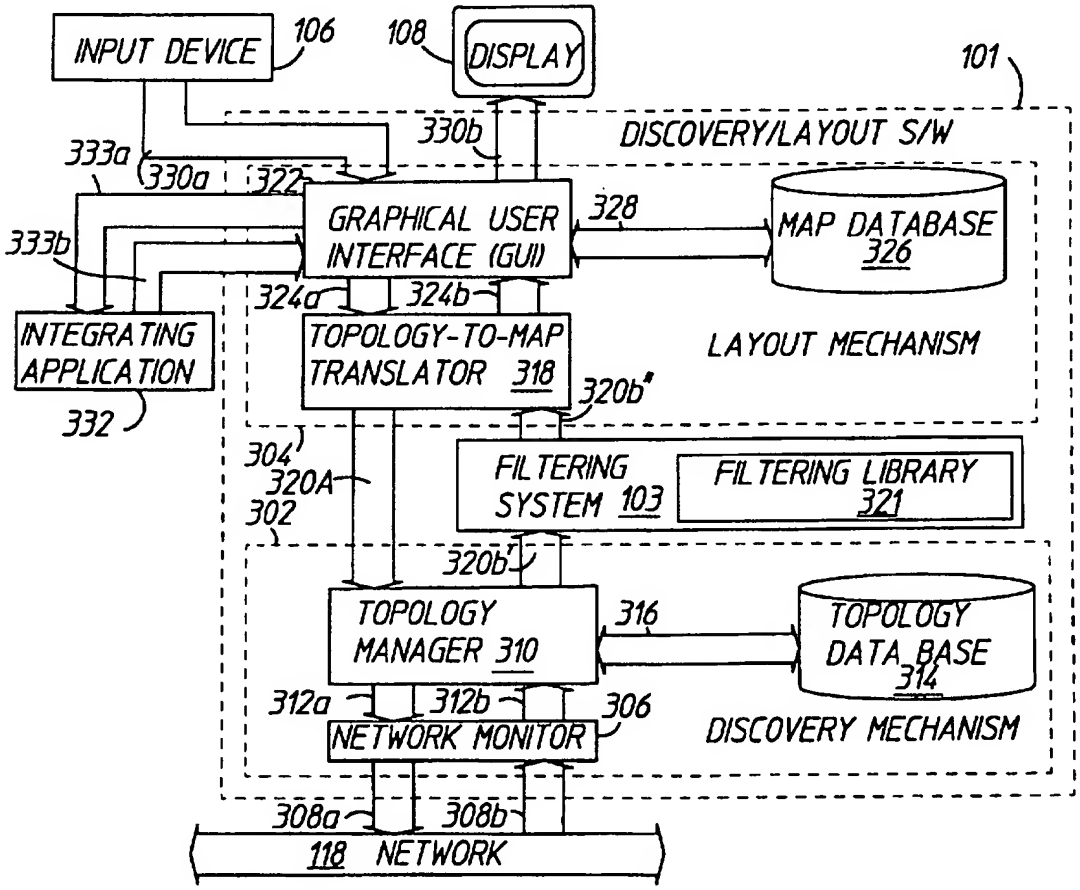
4,107,784	8/1978	Van Bemmelen	345/337
5,021,976	6/1991	Wexelblat et al.	345/356
5,185,860	2/1993	Wu	395/200.54
5,438,659	8/1995	Notess et al.	345/335
5,603,034	2/1997	Swanson	395/701
5,627,979	5/1997	Chang et al.	345/335

Primary Examiner—Tod R. Swann
Assistant Examiner—J. Peikari

[57] **ABSTRACT**

Discovery/layout software configures a general purpose computer system to act as a management station using an industry standard SNMP protocol. The discovery/layout software has a discovery mechanism and a layout mechanism which, in combination, permit the discovery/layout software to provide various submaps to a display for illustrating network topology, which includes devices and device interconnections of a network. The submaps correspond to various hierarchical views of a network. Significantly, one or more filtering systems are provided in the discovery/layout software for filtering objects to be displayed within the submaps. The filtering systems reduce clutter in the submaps, reduce memory usage and associated expense, and reduce interprocess communication (context switching) to achieve higher performance.

31 Claims, 25 Drawing Sheets



2118, which adds the interface pertaining to IFOBJ to the SYMCHANGELIST. If not, then the block 2116 transfers to the block 2120, which adds the interface pertaining to IFOBJ (between node object and network object) to NEWSYMLIST.

At block 2122, the variable NET is set to assume the contents of the network submap 206 (FIG. 2). The contents include, for example but not limited to segments, connections, etc. Block 2122 transfers to block 2124 of FIG. 21B.

With reference to FIG. 21B, at block 2124, a determination is made as to whether SEGOBJ is in NET, i.e., whether the segment object is within the network submap 206 (FIG. 2). If not, then the flow chart terminates. If so, then the block 2124 transfers to the block 2126.

At block 2126, a determination is made as to whether NODEOJ is in NET, i.e., whether the node object is within the network submap 206 (FIG. 2). If not, then the flow chart transfers to block 2134. If so, then the block 2126 transfers to the block 2128.

At block 2128, an inquiry is made as to whether IFOBJ is within NET, i.e., whether the interface object is within the network submap 206 (FIG. 2). If so, then the block 2128 transfers to block 2130, which adds the interface pertaining to IFOBJ to SYMCHANGELIST. If not, then the block 2128 transfers to the block 2132, which adds the interface pertaining to IFOBJ (which is between a node object and a segment object) to NEWSYMLIST. The blocks 2130, 2132 transfer to the block 2134, as is shown in FIG. 21B.

At block 2134, the variable SEG is set to assume the contents of the segment submap 208 (FIG. 2). The contents include, for example but not limited to, nodes and connections. Block 2134 transfers to block 2136.

At block 2136, a determination is made as to whether NODEOBJ is in SEG, i.e., whether the node object is within the segment submap 208 (FIG. 2). If not, then the flow chart transfers to block 2146 of FIG. 21B. If so, then the block 2136 transfers to block 2138.

At block 2138, a determination is made as to whether IFOBJ is within SEG, i.e., whether the interface object is within the segment submap 208 (FIG. 2). If so, then the block 2138 transfers to the block 2142, which adds the interface pertaining to IFOBJ to SYMCHANGELIST. If not, then the block 2138 transfers to the block 2144, which adds the interface pertaining to IFOBJ to NEWSYMLIST. The blocks 2142, 2144 are transferred to the block 2146 of FIG. 21C.

With reference to FIG. 21C, at block 2146, the variable NODE is set to assume the contents of the node submap 210 (FIG. 2). The contents include interface objects. Block 2146 transfers to the block 2148.

At block 2148, a determination is made as to whether IFOBJ is within NODE, i.e., whether the interface object is within the node submap 210 (FIG. 2). If so, then the interface pertaining to IFOBJ is added to SYMCHANGELIST, as indicated at block 2150. If not, then the block 2148 transfers to the block 2152, which adds the interface pertaining to IFOBJ to NEWSYMLIST. Finally, after blocks 2150, 2152, the flow chart contained collectively in FIGS. 21A through 21C terminates.

FIG. 22 shows a flow chart of the architecture and functionality of a preferred embodiment for implementing the update map block 1008 (FIG. 10). In this flow chart, a batch transfer of change is sent by the translator 318 to the GUI 322. With reference to FIG. 22, at block 2202, the translator 318 transfers the NEWSYMLIST to the GUI 322, and in block 2204, the translator 318 transfers the SYMCHANGELIST to the GUI 322. After block 2204, the flow chart of FIG. 22 terminates and the operation passes back to block 1010 (FIG. 10).

FIG. 23 illustrates an on-demand submap module contained within the GUI 322 (FIG. 3). This flow chart implements the user interface to the various submaps of the map 200 (FIG. 2). With reference to FIG. 23, at a block 2302, the GUI 322 monitors the input devices connected to the management station 100 (FIG. 1), for instance, the input device 106. When the user of the management station 100 prompts the management station 100 via the input device 106 or some other input device to explode an object on the display 108, the block 2302 of FIG. 23 transfers to the block 2304 in order to process the user request. At block 2304, a determination is made as to whether the child submap is contained within the map 200 (FIG. 2). If so, then the block 2304 transfers to the block 2308. If not, then the block 2304 transfers to the block 2306, which creates and populates the submap. The GUI 322 populates the submap by requesting the translator 318 to create and populate a submap based on topology data retrieved from the topology manager 310. Moreover, block 2306 transfers to the block 2308 which opens the child submap and displays the child submap on the display 108 for the user.

In concluding the detailed description, it should be noted that it will be obvious to those skilled in the art that many variations and modifications may be made to the preferred embodiments without substantially departing from the principles of the present invention. All such variations and modifications are intended to be included herein within the scope of the present invention, as set forth in the following claims. Further, in the claims hereafter, the structures, materials, acts, and equivalents of all means-plus-function elements or all step-plus-function elements are intended to include any and all structures, materials, or acts for performing the specified functions in combination with the other claimed elements.

Wherefore, the following is claimed:

1. A management system for efficiently discovering and displaying devices and interconnections of a network, comprising:

a processor;

a memory;

a display;

an interface interconnecting said processor, said memory, and said display and capable of connecting to said network;

a discovery mechanism stored in said memory for driving said processor, said discovery mechanism configured to discover and store topology data indicative of said devices and said interconnections of said network;

a layout mechanism stored in said memory for driving said processor, said layout mechanism configured to receive said topology data from said discovery mechanism, said layout mechanism configured to drive said display based upon said topology data; and

a filtering system stored in said memory for driving said processor, said filtering system configured to filter objects within said topology data that pass from said discovery mechanism to said layout mechanism.

2. The system of claim 1, further comprising a library associated with said filtering system, said library configured to specify which of said objects are communicated from said discovery mechanism to said layout mechanism.

3. The system of claim 1, further comprising a second filtering system stored in said memory for driving said processor, said second filtering system configured to filter objects within said topology data that pass from said network to said discovery mechanism.

4. The system of claim 1, further comprising:

a second discovery mechanism stored in said memory for driving said processor, said second discovery mechanism configured to discover and store topology data

indicative of said devices and said interconnections of said network; and

a second filtering system stored in said memory for driving said processor, said second filtering system configured to filter objects within said topology data that pass between said first and second discovery mechanisms.

5. The system of claim 4, further comprising a third filtering system stored in said memory for driving said processor, said third filtering system configured to filter objects within said topology data that pass from said network to said discovery mechanism.

6. The system of claim 5, further comprising a library in communication with said first, second, and third filtering systems, said library configured to specify which of said objects are communicated through said filtering systems.

7. The system of claim 1, wherein said layout mechanism comprises:

a translator configured to convert said topology data to said map data; and

a graphical user interface configured to receive said map data from said translator and to drive said display based upon said map data.

8. The system of claim 5, wherein said translator is configured to generate a plurality of hierarchically arranged submaps from said topology data.

9. The system of claim 8, wherein said hierarchically arranged submaps include an internet submap having at least one network object, at least one network submap associated with said at least one network object and having at least one segment object, at least one segment submap associated with said at least one segment object and having at least one node object, and at least one node submap associated with said at least one node object and having at least one interface object.

10. The system of claim 1, wherein said filtering system includes a boolean expression for determining which of said objects within said topology data pass from said discovery mechanism to said layout mechanism.

11. A management system for efficiently discovering and displaying devices and interconnections of a network, comprising:

a processor;

a memory;

a display;

an interface interconnecting said processor, said memory, and said display and capable of connecting to said network;

a discovery mechanism stored in said memory for driving said processor, said discovery mechanism configured to discover and store topology data indicative of said devices and said interconnections of said network;

a layout mechanism stored in said memory for driving said processor, said layout mechanism configured to receive said topology data from said discovery mechanism, said layout mechanism configured to drive said display based upon said topology data; and

a filtering system stored in said memory for driving said processor, said filtering system configured to filter objects within said topology data that pass from said network to said discovery mechanism.

12. The system of claim 11, further comprising a library associated with said filtering system, said library configured to specify which of said objects are communicated from said network to said discovery mechanism.

13. The system of claim 11, wherein said layout mechanism comprises:

a translator configured to convert said topology data to said map data; and

a graphical user interface configured to receive said map data from said translator and to drive said display based upon said map data.

14. The system of claim 13, wherein said translator is configured to generate a plurality of hierarchically arranged submaps from said topology data.

15. The system of claim 14, wherein said hierarchically arranged submaps include an internet submap having at least one network object, at least one network submap associated with said at least one network object and having at least one segment object, at least one segment submap associated with said at least one segment object and having at least one node object, and at least one node submap associated with said at least one node object and having at least one interface object.

16. The system of claim 11, wherein said filtering system includes a boolean expression for determining which of said objects within said topology data pass from said network to said discovery mechanism.

17. A management system for efficiently discovering and displaying devices and interconnections of a network, comprising:

a processor;

a memory;

a display;

an interface interconnecting said processor, said memory, and said display and capable of connecting to said network;

first and second discovery mechanisms stored in said memory for driving said processor, said first and second discovery mechanisms configured to discover and store topology data indicative of said devices and said interconnections of said network;

a layout mechanism stored in said memory for driving said processor, said layout mechanism configured to receive said topology data from said discovery mechanism, said layout mechanism configured to drive said display based upon said topology data; and

a filtering system stored in said memory for driving said processor, said filtering system configured to filter objects within said topology data that pass between said first and second discovery mechanisms.

18. The system of claim 17, further comprising a library associated with said filtering system, said library configured to specify which of said objects are communicated between said first and second discovery mechanisms.

19. The system of claim 17, wherein said layout mechanism comprises:

a translator configured to convert said topology data to said map data; and

a graphical user interface configured to receive said map data from said translator and to drive said display based upon said map data.

20. The system of claim 19, wherein said translator is configured to generate a plurality of hierarchically arranged submaps from said topology data.

21. The system of claim 20, wherein said hierarchically arranged submaps include an internet submap having at least one network object, at least one network submap associated with said at least one network object and having at least one segment object, at least one segment submap associated with said at least one segment object and having at least one node object, and at least one node submap associated with said at least one node object and having at least one interface object.

22. The system of claim 17, wherein said filtering system includes a boolean expression for determining which of said objects within said topology data pass between said first and second discovery mechanisms.

23. A filtering system for discovering and displaying devices and interconnections of a network, comprising:

means for generating topology data indicative of said devices and said interconnections of said network;

means for comparing objects within said topology data with a predefined library to determine allowable objects and nonallowable objects;

means for converting said allowable objects into map data and displaying said map data; and

means for refraining from converting said nonallowable objects into map data and from displaying said map data.

24. A filtering system for discovering and displaying devices and interconnections of a network, comprising:

means for generating topology data indicative of said devices and said interconnections of said network;

means for comparing objects within said topology data with a predefined library to determine allowable objects and nonallowable objects;

means for converting said allowable objects into map data and displaying said map data; and

means for eliminating said nonallowable objects from said topology data.

25. A filtering system for discovering and displaying devices and interconnections of a network, comprising:

first and second discovery means for generating topology data indicative of said devices and said interconnections of said network;

means for communicating objects within said topology data between said first and second discovery means;

means for comparing said objects that are communicated between said first and second discovery means with a predefined library to determine allowable objects and nonallowable objects; and

means for preventing transfer of said nonallowable objects between said first and second discovery means, while permitting transfer of said allowable objects therebetween.

26. A computer-readable medium in a management station comprising a program for discovering and displaying devices and interconnections of a network, said program comprising:

a discovery mechanism configured to discover and store topology data indicative of said devices and said interconnections of said network;

a layout mechanism configured to receive said topology data from said discovery mechanism, said layout mechanism configured to drive said display based upon said topology data; and

a filtering system configured to filter objects within :said topology data that pass from said discovery mechanism to said layout mechanism.

27. A computer-readable medium in a management station comprising a program for discovering and displaying devices and interconnections of a network, said program comprising:

a discovery mechanism configured to discover and store topology data indicative of said devices and said interconnections of said network;

a layout mechanism configured to receive said topology data from said discovery mechanism, said layout mechanism configured to drive said display based upon said topology data; and

a filtering system configured to filter objects within said topology data that pass from said network to said discovery mechanism.

28. A computer-readable medium in a management station comprising a program for discovering and displaying devices and interconnections of a network, said program comprising:

first and second discovery mechanisms configured to discover and store topology data indicative of said devices and said interconnections of said network;

a layout mechanism configured to receive said topology data from said first and second discovery mechanisms, said layout mechanism configured to drive said display based upon said topology data; and

a filtering system configured to filter objects within said topology data that pass between said first and second discovery mechanisms.

29. A filtering method for discovering and displaying devices and interconnections of a network, comprising the steps of:

generating topology data indicative of said devices and said interconnections of said network;

comparing objects within said topology data with a predefined library to determine allowable objects and nonallowable objects;

converting said allowable objects into map data and displaying said map data; and

refraining from converting said nonallowable objects into map data and from displaying said map data.

30. A filtering method for discovering and displaying devices and interconnections of a network, comprising the steps of:

generating topology data indicative of said devices and said interconnections of said network;

comparing objects within said topology data with a predefined library to determine allowable objects and nonallowable objects;

converting said allowable objects into map data and displaying said map data; and

eliminating said nonallowable objects from said topology data.

31. A filtering method for discovering and displaying devices and interconnections of a network, comprising the steps of:

generating topology data indicative of said devices and said interconnections of said network with first and second discovery mechanisms that are interfaced to said network;

communicating objects within said topology data between said first and second discovery mechanisms;

comparing said objects that are communicated between said first and second discovery mechanisms with a predefined library to determine allowable objects and nonallowable objects; and

preventing transfer of said nonallowable objects between said first and second discovery mechanisms, while permitting transfer of said allowable objects therebetween.

* * * * *

58/3,K/84 (Item 84 from file: 348) **Links**
EUROPEAN PATENTS

01046870
Object-oriented system for mapping structered information to different structured information
Objekt orientiertes System um strukturierte Informationen in andere strukturierte Informationen umzuwandeln
Systeme oriente objet pour transformer des Informations structurees en autres informations structurees
PATENT ASSIGNEE:
Ricoh Company, Ltd., (3895230), 3-6, Nakamagome 1-chome, Ohta-ku, Tokyo 143-8555, (JP), (Applicant designated States: all)
INVENTOR:
Motoyama, Tetsuro, c/o RICOH CORPORATION, Systems R & D Group, 3001 Orchard Parkway, San Jose, CA 95134-2088, (US)
Fong, Avery, c/o RICOH CORPORATION, Systems R & D Group, 3001 Orchard Parkway, San Jose, CA 95134-2088, (US)
Bhatnagar,Anurag, c/o RICOH CORPORATION, Systems R & D Group, 3001 Orchard Parkway, San Jose, CA 95134-2088, (US)
LEGAL REPRESENTATIVE:
Schwabe - Sandmair - Marx (100951), Stuntzstrasse 16, 81677 Munchen, (DE)
PATENT (CC, No, Kind, Date): EP 926607 A2 990630 (Basic)
EP 926607 A3 060118
APPLICATION (CC, No, Date): EP 98124276 981218;
PRIORITY (CC, No, Date): US 997482 971223; US 997705 971223
DESIGNATED STATES: AT; BE; CH; CY; DE; DK; ES; FI; FR; GB; GR; IE; IT; LI; LU; MC; NL; PT; SE
EXTENDED DESIGNATED STATES: AL; LT; LV; MK; RO; SI
INTERNATIONAL PATENT CLASS (V7): G06F-017/30;
INTERNATIONAL CLASSIFICATION (V8 + ATTRIBUTES):
IPC + Level Value Position Status Version Action Source Office:
G06F-0017/30 A I F B 20060101 19990402 H EP
ABSTRACT WORD COUNT: 246
NOTE:
Figure number on first page: 5

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	199926	7812
SPEC A	(English)	199926	17873
Total word count - document A			25689
Total word count - document B			0
Total word count - documents A + B			25689

INTERNATIONAL PATENT CLASS (V7): G06F-017/30
INTERNATIONAL CLASSIFICATION (V8 + ATTRIBUTES):
IPC + Level Value Position Status Version Action Source Office:
G06F-0017/30 A I F B 20060101 19990402 H EP

...SPECIFICATION SGML DTDs to HTML DTDs. The problem of DTD to DTD mapping is the default mapping from an SGML instance to an HTML instance based

upon the DTDs. SGML tags are either mapped to zero or more HTML tags, and the sources of HTML attributes must be specified in the mapping. A more concise mathematical expression of the problem is given below.

Let SS be the...

...H1))>,..., <H0)), H1))>, <H0)), H2))>,..., <H1)), H0))>,...). The sequence of legal HTML tags to be mapped are likely to be found in HG. Then the SGML tag to HTML tag mapping is equivalent to the function F : SS --> HG + (not-assigned).

For purposes of this discussion...

...Si))). Assume H0)) has Attr0)) and Attr1)), H6)) has Attr0)), and S0)) and S1)) are mapped to <H0))> and <H3)), H6))>, respectively. Then HGG=(null, <S0)), <H0))>, <H0))Tag Name, Attr0...

...tag name and one attribute name, and AC=the set of tag names with character data content. Then the identification of the attribute source is a function G mapping from HGG to Satr, denoted G: HGG-->SAtr.

A complete description of SGML is provided...

...of Fig. 1C, together with the exemplary SGML DTD of Fig. 1A, and the exemplary mapping of Fig. 1B are utilized in a transformation process to generate the HTML document of...

...the SGML document of Fig. 1C. These components are then utilized in conjunction with the map of Fig. 1B to transform the SGML document of Fig. 1C into the HTML document...

...replaced by other characters.

Line 28 of Fig. 1A is a declaration for an attribute list for the element t1. An attribute is a property of an element that takes on...

...values for different instances of elements. For example, an element 'person' typically has an attribute list of attributes 'name', 'age', and 'haircolor'. A particular first person has name="Joe Smith", age...

...of a different value. For the example of Fig. 1A, on line 28, the attribute list includes an attribute 'name', of type CDATA. The character string '#REQUIRED' is an attribute value...

...is a declaration of an element t4, of type CDATA.

In the SGML to HTML mapping of Fig. 1B, line 42 illustrates a mapping rule of the element t of line 24 to a string of HTML tags and text including '<html><title>Title</title>'. Line 44 illustrates a mapping rule of the element t1 of line 26 to a string of HTML tags '

<A...

...of line 44 is a rule rather than an attribute value. Line 46 illustrates a mapping rule of the element t2 of line 30 to an HTML tag '

'. Line 48 illustrates a mapping rule of the element t3 of line 32 to an HTML tag '

'. Line 50 illustrates a mapping rule of the element t4 of line 34 to a

string of HTML tags '
<A...

...the output of the transformation process utilizing the SGML DTD file of Fig. 1A, the mapping of Fig. 1B, and the SGML document of Fig. 1C. Lines 82, 84, 86, 88...

US006085196A

United States Patent [19]

Motoyama et al.

[11] Patent Number: 6,085,196

[45] Date of Patent: Jul. 4, 2000

[54] OBJECT-ORIENTED SYSTEM AND COMPUTER PROGRAM PRODUCT FOR MAPPING STRUCTURED INFORMATION TO DIFFERENT STRUCTURED INFORMATION

[75] Inventors: Tetsuro Motoyama, Cupertino; Avery Fong, Hayward; Anurag Bhatnagar, Sunnyvale, all of Calif.

[73] Assignees: Ricoh Company, Ltd., Tokyo, Japan; Ricoh Corporation, San Jose, Calif.

[21] Appl. No.: 08/997,482

[22] Filed: Dec. 23, 1997

[51] Int. Cl.[7] G06F 17/30

[52] U.S. Cl. 707/102

[58] Field of Search 707/102

[56] References Cited

U.S. PATENT DOCUMENTS

5,530,852	6/1996	Meske, Jr. et al.	709/206
5,832,496	11/1998	Anand et al.	707/102
5,870,746	2/1999	Knutson et al.	707/101
5,895,476	4/1999	Orr et al.	707/517
5,907,704	5/1999	Gudmundson et al.	395/701
5,909,684	6/1999	Nelson	707/103
5,913,065	6/1999	Faustini	395/703
5,928,323	7/1999	Gosling et al.	709/203
5,944,783	8/1999	Nieten	709/202
5,956,737	9/1999	King et al.	707/517
6,012,066	8/1999	Discount et al.	707/143

OTHER PUBLICATIONS

OmniMark LE Description (visited Oct. 6, 1997) <http://www.omnimark.com/release/omle/31/omlesamp.htm>.

Robin Cover. Public SGML Software (last modified Sep. 23, 1997) <http://www.sil.org/sgml/publicSW.html>.

Robin Cover. SGML: First Freeware SGML Viewer for the World Wide Web (last modified Jul. 11, 1997) <http://www.sil.org/sgml/free-pan.html>.

dtd2html (visited Oct. 6, 1997) <http://www.oac.uci.edu/indiv/ehood/perlSGML/doc/html/dtd2html.html>.

Grey Matter (visited Oct. 6, 1997) <http://www.greymatter-.co.uk/grmWEB/items/00000370.htm>.

(List continued on next page.)

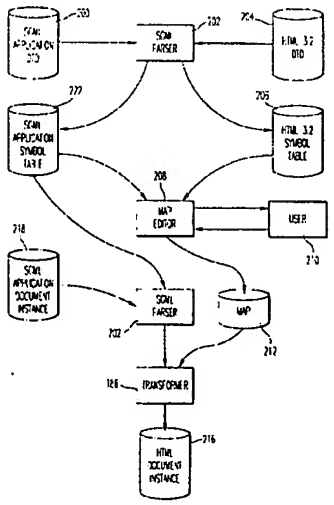
Primary Examiner—Wayne Amsbury

Attorney, Agent, or Firm—Oblon, Spivak, McClelland, Maier & Neustadt, P.C.

[57] ABSTRACT

An object-oriented system and computer program product for mapping structured information to different structured information, which allows a user to interactively define the mapping. The present invention operates as an object-oriented user tool by accepting interactive input from a user of a source input, by processing the input to display the source input in a format for accepting and processing user commands to create or edit a transformation map of source components to target components. Interactive user input is then accepted and processed for selection of an input file to be transformed and selection of a transformation map to be used for the requested transformation. Interactive user input is accepted and processed for selection of individual components of the first structured information format for mapping, and for selection of options for the target components. Exemplary options for the target components are a null value, the source component itself, a single selected target component, or plural selected target components. Interactive user input is accepted for processing to assign attribute values to components of the second structured information format. Exemplary options for the sources of attribute values are attribute values obtained from the source components, system attribute values, no value, attribute values input interactively by the user, and content of element. Interactive user input is then accepted and processed to initiate processing of a transformation of the source input file in the first structured information format to a target output file in the second structured information format.

31 Claims, 38 Drawing Sheets



allows the user to create a new directory name in the UNIX file system window **1504**. The map option **1508** allows the user to request that a map be created at the time of request. The merge all **1510** option allows the user to create a UNIX file name **1504** by merging all the components of the public identifier name **1502** into a flat file name **1504**. The next **1512** option allows the user to step to a next screen. The previous **1514** option allows the user to back up to the previous screen.

FIG. **20E** illustrates an exemplary user interface **1520** for a registered owner **1522** component of the ISO/IEC 9070 public identifier **1502** of FIG. **20D**. User interface **1520** options presented are a window **1522** showing a prefix and an owner-name component. User options are a map individually **1524**, a merge both **1526**, a next **1527**, and a previous **1528**. The map individually **1524** option allows the user to map individual components of the ISO/IEC 9070 name **1522** to individual components of the UNIX file system scheme **1504** of FIG. **20D**. The merge both **1526** option allows the user to merge components of the registered owner name **1522** into one flat UNIX file name or directory name. The next **1527** option allows the user to step to a next screen. The previous **1528** option allows the user to back up to the previous screen.

FIG. **20F** illustrates an exemplary user interface **1530** for mapping the prefix, owner name component separator to the UNIX legal character set format **1534**. The registered owner component has a prefix and an owner-name component separator "::" **1532** which is not a widely used character string in the UNIX environment. The user is allowed to map the '::' separator **1532** to any of the valid characters in the UNIX file system character set **1534**, with a mapping to '_' as a default mapping. User options are a map **1536**, a next **1537**, and a previous **1538**. The map **1536** option allows the user to select creating a map, with the assumption that the user has finished selecting options for creation of the map. The next **1537** option allows the user to step to a next screen. The previous **1538** option allows the user to back up to the previous screen.

FIG. **20G** illustrates an exemplary user interface **1540** for mapping an owner name character **1542** to valid characters **1544** of the UNIX file system format. The user is given the options of mapping special characters **1542** which are valid in the ISO/IEC 9070 scheme to characters which are valid in the UNIX file system scheme **1544**. A mapping of a character in the ISO/IEC 9070 scheme **1542** is set to '_' as a default mapping. The user is given options of a map **1546**, a next **1548**, and a previous **1550**. The map **1546** option allows the user to select creating a map, with the assumption that the user has finished selecting options for creation of the map. The next **1548** option allows the user to step to a next screen. The previous **1550** option allows the user to back up to the previous screen.

FIG. **20H** illustrates an exemplary user interface **1560** for the user to map a registered owner component **1562** to a UNIX file scheme format **1564**. The user is allowed to select a prefix component of the registered owner name or an owner name component other than prefix **1562**. The user is allowed to select a directory option in the UNIX scheme **1564**. The user is also allowed to select a file with object name option in the UNIX file scheme **1564**. The user is given an option of a map **1566** for creating the map with the options currently selected. The user is also given an option of a previous **1568** to back up to the previous screen.

The present invention has been described using an exemplary implementation of a mapping creator and editor for an SGML to HTML transformer with user interaction for creation and editing of the map, and an exemplary mapping creator and editor for an ISO/IEC 9070 to a UNIX file format transformer. The example shown in this disclosure uses OOP and Windows GUI techniques to implement the user interface, map processing, and transformation. However, the user interface can be implemented using text line queries or menus. Programming methodologies other than OOP can be used for implementing the processing. References to storage areas can be made by techniques other than using pointers.

This invention may be conveniently implemented using a conventional general purpose digital computer or microprocessor programmed according to the teachings of the present specification, as will be apparent to those skilled in the computer art. Appropriate software coding can readily be prepared by skilled programmers based on the teachings of the present disclosure, as will be apparent to those skilled in the software art. The invention may also be implemented by the preparation of application specific integrated circuits or by interconnecting an appropriate network of conventional component circuits, as will be readily apparent to those skilled in the art.

The present invention includes a computer program product which is a storage medium including instructions which can be used to program a computer to perform a process of the invention. The storage medium can include, but is not limited to, any type of disk including floppy disks, optical discs, CD-ROMs, and magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, magnetic or optical cards, or any type of media suitable for storing electronic instructions.

Obviously, numerous modifications and variations of the present invention are possible in light of the above teachings. It is therefore to be understood that within the scope of the appended claims, the invention may be practiced otherwise than as specifically described herein.

What is claimed as new and desired to be secured by Letters Patent of the United States is:

1. An object-oriented system for processing structured information for implementation by a computer in an object-oriented framework, comprising:

a storage means;

a first obtaining means for obtaining an interactive input from a user;

a second obtaining means for obtaining a first structural description of a first structured information format;

a third obtaining means for obtaining a second structural description of a second structured information format;

means for interactively creating a rule to transform an element of the first structured information format into an element of the second structured information format utilizing the interactive input from the user, the first structural description, and the second structural description; and

means for outputting the rule,

wherein at least one of the first obtaining means, the second obtaining means, the third obtaining means, the means for interactively creating, and the means for outputting includes a software object.

2. A system according to claim **1**, wherein the first structured information format includes an ISO/IEC 9070 public identifier naming format, the second structured information format includes an operating system file name format, and the means for interactively creating comprises:

means for interactively creating a rule to transform an ISO/IEC 9070 public identifier naming format element

into an operating system file name format element utilizing the interactive input from the user, a structural description of the ISO/IEC 9070 public identifier naming format, and a structural description of the operating system file name format.

3. A system according to claim 1, wherein the first structured information format includes a first database variable name format, the second structured information format includes a second database variable name format, and the means for interactively creating comprises:

means for interactively creating a rule to transform a first database variable name format element into a second database variable name format element utilizing the interactive input from the user, a structural description of the first database variable name format, and a structural description of the second database variable name format.

4. A system according to claim 1, wherein the structured information includes a markup language format, the first structured information format includes a first markup language format, the second structured information format includes a second markup language format, and the means for interactively creating comprises:

means for interactively creating a rule to transform a first markup language format element into a second markup language format element utilizing the interactive input from the user, a structural description of the first markup language format, and a structural description of the second markup language format.

5. A system according to claim 4, wherein the first markup language format includes a Standard Generalized Markup Language ("SGML"), the second markup language format includes a HyperText Markup Language ("HTML"), and the means for interactively creating comprises:

means for interactively creating a rule to transform an SGML element of the first markup language format into an HTML element of the second markup language format utilizing the interactive input from the user, the first structural description which includes an SGML Document Type Definition ("DTD"), and the second structural description which includes an HTML DTD.

6. A system according to claim 1, wherein the means for interactively creating comprises:

a map creator object.

7. A system according to claim 6, wherein the means for outputting the rule comprises:

an object method for outputting the rule to a map object.

8. A system according to claim 6, wherein the map creator object comprises:

a reference to a software object for an element for transformation of the first structured information format;

a reference to a software object for an element of the second structured information format, for transformation of the element of the first structured information format;

a reference to a software object for a property of the element of the second structured information format, for transformation of the element of the first structured information format;

a reference to a software object for an attribute value of the element of the second structured information format, for transformation of the element of the first structured information format;

an object method for obtaining the element for transformation of the first structured information format, which has been interactively selected by the user, using the software object for the element for transformation of the first structured information format;

an object method for obtaining the element of the second structured information format which corresponds to the element of the first structured information format, which has been interactively selected by the user, using the software object for the element of the second structured information format;

an object method for determining a property of the element of the second structured information format which has been selected by the user, using the software object for a property of the element of the second structured information format;

an object method for obtaining a second structured information format attribute value which has been interactively input by a user, using the software object for the attribute value of the element of the second structured information format; and

an object method for assigning the attribute value which has been interactively input by a user to the second structured information format attribute value.

9. A system according to claim 8, wherein the map creator object further comprises:

a reference to a software object for registering an instance of an element for transformation of the first structured information format; and

a reference to a software object for unregistering the instance of an element for transformation of the first structured information format when the element is no longer needed by the map creator.

10. A system according to claim 1, further comprising:

means for processing an element of the first structured information format into a plurality of first structured information format components.

11. A system according to claim 10, wherein the means for processing an element of the first structured information format into a plurality of first structured information format components comprises:

a parser object.

12. A system according to claim 11, wherein the parser object comprises:

a reference to the element of the first structured information format;

a reference to a storage area for the plurality of first structured information format components; and

an object method for processing the element of the first structured information format into the plurality of first structured information format components for storage in the storage area for the plurality of first structured information format components.

13. A system according to claim 12, further comprising:

means for utilizing the rule to transform the element of the first structured information format into the element of the second structured information format.

14. A system according to claim 13, wherein the means for utilizing the rule to transform the element of the first structured information format into the element of the second structured information format comprises:

a transformer object.

15. A system according to claim 1, wherein the means for obtaining an interactive input from a user comprises:

a user interface object.

16. A system according to claim 15, wherein the user interface object comprises:

a reference to a software object for user input;

an object method for obtaining interactive input from the user of a selection of an element for transformation of a first structured information format using the software object for user input; and

an object method for obtaining interactive input from the user of a selection of an element of a second structured information format which corresponds to the element of the first structured information format using the software object for user input.

17. A system according to claim **15**, wherein the user interface object further comprises:

a reference to a software object for user input of a selection of a transformation to be performed on the element of the first structured information format; and

an object method for obtaining interactive input from the user of the selection of a transformation to be performed on the element of the first structured information format using the software object for user input of the selection of the transformation.

18. A system according to claim **17**, wherein the means for interactively creating a rule to transform an element of a first structured information format into an element of a second structured information format utilizing the interactive input from the user further comprises:

a reference to a software object for a rule to be created; and

an object method for creating a rule to map the second element of the first structured information format to a null string using the software object for the rule to be created, when the selection of a transformation which has been input by the user indicates a null transformation is to be performed.

19. A system according to claim **17**, wherein the means for interactively creating a rule to transform an element of a first structured information format into an element of a second structured information format utilizing the interactive input from the user further comprises:

a reference to a software object for a rule to be created; and

an object method for creating a rule to map the second element of the first structured information format to a copy of the second element of the first structured information format using the software object for the rule to be created, when the selection of a transformation which has been input by the user indicates a transformation of the second element of the first structured information format to a copy of the second element of the first structured information format is to be performed.

20. A system according to claim **15**, wherein the user interface object further comprises:

a reference to a software object for an interactive user input of a source for inputting the second structured information format attribute value;

a reference to a software object for an interactive user input of the second structured information format attribute value;

an object method for obtaining interactive input from the user of the source for inputting the second structured information format attribute value using the software object for the interactive user input of the source for inputting the second structured information format attribute value; and

an object method for obtaining interactive input from the user of the second structured information format attribute value using the software object for the interactive user input of the second structured information format attribute value.

21. A system according to claim **20**, further comprising:

a reference to a software object for a rule to be created;

an object method for examining the source which has been input by the user; and

an object method for assigning a null value to the second structured information format attribute value using the software object for the rule to be created, when the source which has been input by the user indicates no source is to be used.

22. A system according to claim **20**, further comprising:

a reference to a software object for a rule to be created;

an object method for examining the source which has been input by the user; and

an object method for assigning a system value to the second structured information format attribute value using the software object for the rule to be created, when the source which has been input by the user indicates a system source is to be used.

23. A system according to claim **20**, further comprising:

a reference to a software object for a rule to be created;

an object method for examining the source which has been input by the user; and

an object method for assigning a first structured information format attribute value to the second structured information format attribute value using the software object for the rule to be created, when the source which has been input by the user indicates a first structured information format attribute source is to be used.

24. A system according to claim **20**, further comprising:

a reference to a software object for a rule to be created;

an object method for examining the source which has been input by the user; and

an object method for assigning a first structured information format content value to the second structured information format attribute value using the software object for the rule to be created, when the source which has been input by the user indicates a first structured information format content source is to be used.

25. A system according to claim **20**, further comprising:

a reference to a storage buffer for the source which has been input by the user;

an object method for examining the source which has been input by the user using the storage buffer for the source which has been input by the user;

an object method for interactively inputting a user input value, when the source which has been input by the user indicates a user input source is to be used; and

an object method for assigning the user input value to the second structured information format attribute value, when the source which has been input by the user indicates a user input source is to be used.

26. A system according to claim **1**, wherein the user comprises:

a software object.

27. An object-oriented computer program product for processing structured information for implementation by a computer in an object-oriented framework, comprising:

a storage means;

a first obtaining means for obtaining an interactive input from a user;

a second obtaining means for obtaining a first structural description of a first structured information format;

a third obtaining means for obtaining a second structural description of a second structured information format;

means for interactively creating a rule to transform an element of the first structured information format into an element of the second structured information format utilizing the interactive input from the user, the first structural description, and the second structural description; and

means for outputting the rule,

wherein at least one of the first obtaining means, the second obtaining means, the third obtaining means, the means for interactively creating, and the means for outputting includes a software object.

28. A computer program product according to claim **27**, wherein the first structured information format includes ISO/IEC 9070 public identifier naming format, the second structured information format includes an operating system file name format and the means for interactively creating comprises:

means for interactively creating a rule to transform an element of a first structured information format which includes an ISO/IEC 9070 public identifier element into an element of a second structured information format which includes an operating system file name format element utilizing the interactive input from the user, the first structural description which includes a structural description of the ISO/IEC 9070 public identifier format, and the second structural description which includes a structural description of the operating system file name format.

29. A computer program product according to claim **27**, wherein the structured information includes database variable names, the first structured information format includes a first database variable name format, the second structured information format includes a second database variable name format, and the means for interactively comprises:

means for interactively creating a rule to transform an element of a first structured information format which includes a first database variable name format element into an element of a second structured information format which includes a second database variable name format element utilizing the interactive input from the user, the first structural description which includes a structural description of the first database variable name format, and the second structural description which includes a structural description of the second database variable name format.

30. A computer program product according to claim **27**, wherein the structured information includes markup language, the first structured information format includes a first markup language, the second structured information format includes a second markup language, and the means for interactively comprises:

means for interactively a rule to transform an element of a first structured information format which includes a first markup language element into an element of a second structured information format which includes a second markup language element utilizing the interactive input from the user, the first structural description which includes a structural description of the first markup language, and the second structural description which includes a structural description of the second markup language.

31. A computer program product according to claim **30**, wherein the first markup language includes SGML, the second markup language includes HTML, and the means for interactively further comprises:

means for interactively a rule to transform an element of a first markup language which includes an SGML element into an element of a second markup language which includes an HTML element utilizing the interactive input from the user, the first structural description which includes an SGML DTD, and the second structural description which includes an HTML DTD.

* * * * *

58/3,K/46 (Item 46 from file: 348) **Links**
EUROPEAN PATENTS

00858026
METHOD FOR MANAGING DYNAMIC RELATIONS BETWEEN OBJECTS IN DYNAMIC OBJECT-ORIENTED LANGUAGES
VERFAHREN ZUR VERWALTUNG DYNAMISCHER RELATIONEN ZWISCHEN OBJEKTEN IN DYNAMISCH OBJEKTORIENTIERTEN PROGRAMMIERSPRACHEN
PROCEDE DE GESTION DE RELATIONS DYNAMIQUES ENTRE OBJETS DANS DES LANGAGES DYNAMIQUES ORIENTES OBJETS
PATENT ASSIGNEE:
Viewsoft, Inc., (2322080), Suite 300, 250 West Center Street, Provo, UT 84601, (US), (Proprietor designated states: all)
INVENTOR:
MITCHELL, David, C., 1823 South 250, South Orem, UT 84058, (US)
ANDERSON, Kelly, L., 1962 East California Avenue, Provo, UT 84606, (US)
OSMAN, Andrew, V., 231 South 100 East, Provo, UT 84606, (US)
MITCHELL, Dale, K., 361 North 300 West 303, Provo, UT 84601, (US)
LEGAL REPRESENTATIVE:
Horner, David Richard et al (77632), D Young & Co, 21 New Fetter Lane , London EC4A 1DA, (GB)
PATENT (CC, No, Kind, Date): EP 857331 A1 980812 (Basic)
EP 857331 B1 000726
WO 9715883 970501
APPLICATION (CC, No, Date): EP 96936844 961022; WO 96US16927 961022
PRIORITY (CC, No, Date): US 548536 951026
DESIGNATED STATES: AT; BE; CH; DE; FR; GB; IT; LI
INTERNATIONAL PATENT CLASS (V7): G06F-009/46
NOTE:
No A-document published by EPO
LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	200030	604
CLAIMS B	(German)	200030	635
CLAIMS B	(French)	200030	680
SPEC B	(English)	200030	17571
Total word count - document A			0
Total word count - document B			19490
Total word count - documents A + B			19490

INTERNATIONAL PATENT CLASS (V7): G06F-009/46

...SPECIFICATION than leftHandSideChanged in order to do transformations without regards to direction. The parameters are: The field that is changing, the type element that is changing (in this case the left side (107)), and the type element for the other side (104). The type element may represent a field, function, property or a list of members.

void EosMapElement::valueChanged(EosProbeObject *src,EosTypeElement *srcEI,EosTypeElement *destEI);
In the simple case of the field to field mapper, valueChanged simply gets the value from the source side parameter (that changed) and sets the

value on the destination side to this new value through the right type element (104).

6. The right type element (104) sets the field (105) in the right hand object (106) through the dynamic binding mechanism. The type element has already precomputed the index of the field from the field name, so that this is as efficient as possible.

7. Probes now fire on the...

...the mapper to be called. (Again, in an alternative embodiment this could be in the field element (104) rather than directly in the mapper) This callback again checks to see if the mapper active flag is set, and since it is, processing stops; that is, the value is not forwarded to the left hand field (101). This check helps to avoid infinite looping within the mapping machinery. (The probes mechanism also has circular reduction machinery, but adding it to the mappers is more efficient and safer.) Of course this behavior can be overridden in cases where it makes sense to do so.

The probes must be allowed to fire on the field (105) because other objects may have planted probes on this value and they would not be synchronized if the probe did not fire. If the right hand field (105) changes first, then the rightHandSideCallback would have forwarded the information to valueChanged, however the parameters would...

...details of the implementation of valueChanged, leftHandSideChanged and rightHandSideChanged may of course change for each mapper subclass. The general mechanism is flexible enough to handle most of the interesting cases. In this simple case, the mapper simply passed along the value, unchanged, from the left hand patron object to the right hand patron object. It should be noted, however, that the mapper could do anything with the value. It could convert it from metric units to English units, it could use the number as an index into a database and pass a field from record N on to the second patron object. Anything that can be done in a robust programming language can be done inside of the mapper. If a mapper is bi-directional, however, the transformation should always be transitive. That is, the transformation should...

...of the generality obtained comes from the API of the base class EosMapElement for the mappers, it also comes from the type elements that represent the member or members involved with...

...semantic link. As shown in figure one, there is a single type element for each object being connected to. If multiple members are mapped to, the type element holds a list of other type elements.

Network Considerations

Figure 2 shows the same semantic link as figure 1 with the right hand patron object (206) on a second machine.

Note that either or both fields in the patron objects in figure 1 (101) or (105) could be replaced by a network proxy object. When the patron object (106) is actually a network proxy object (206) the details change slightly. Essentially, the proxy object pretends to be the object that it represents. A generic proxy such as this cannot handle normal non-dynamic language syntax, such as invoking a function directly off of the object, but what it can do is provide the services of probing and dynamic binding

as if it were the patron object. This is because dynamic binding and probing are the same for all classes in the language.

The type element sets the data on the proxy object in the normal way through dynamic binding, but the proxy object takes care of packaging and sending the message to the real object over the network. In the preferred embodiment, the network protocols were implemented using a commonly available RPC based networking library, they have an identifier for the object to send the message to, the data that changed and the path to the field of the element to set. Dynamic binding is able to get and set fields by index, or invoke functions by index, so the proxy object must be able to deal with these three forms of information as it builds up the RPC call to the other side of the network. Different mappers generate these different types of information, so they are all put into a canonical form...

...implementation are beyond the scope of this disclosure, but are commonly available.

EosProxy

The proxy object is constructed with a connection object, the object to point to the root object involved in the connection, a path to get to the subfield (described below), and the name of the field that it represents.

EosProxyProbeObject(EosConnection *connection,EosObject *object,const EosObjectPath &path,const EosAtom &name); As each proxy is set up, a unique identifier is generated for the object instance on both sides of the network. The id on ...so the id is passed to the other machine during set up, so that the object will be accessible later. Both machines have a lookup table that is used to transform the ID into the actual pointer in the address space of that machine. The proxy object also must reset itself whenever elements in the subclass path change. An exemplary implementation of the proxy object is given in Appendix I. Note that there is a probe object client and a probe object server. The only difference is that the client initializes the link. After instantiation, a bi...

...the client back passing the server ID number back. The details of initially distributing the objects over the network are beyond the scope of this disclosure.

US005872973A

United States Patent [19]

Mitchell et al.

[11] Patent Number: 5,872,973

[45] Date of Patent: *Feb. 16, 1999

[54] METHOD FOR MANAGING DYNAMIC RELATIONS BETWEEN OBJECTS IN DYNAMIC OBJECT-ORIENTED LANGUAGES

[75] Inventors: David C. Mitchell, South Orem; Kelly L. Anderson, Provo; Andrew V. Osman, Provo; Dale K. Mitchell, Provo, all of Utah

[73] Assignee: Viewsoft, Inc., Provo, Utah

[*] Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

[21] Appl. No.: 548,536

[22] Filed: Oct. 26, 1995

[51] Int. Cl.[6] G06F 9/44

[52] U.S. Cl. 395/685; 395/683; 395/702

[58] Field of Search 395/685, 710, 395/614, 683, 701, 702, 708; 364/284.4; 707/103

[56] References Cited

U.S. PATENT DOCUMENTS

5,327,562	7/1994	Adcock	395/700
5,339,430	8/1994	Lundin et al.	395/700
5,359,721	10/1994	Kempf et al.	395/425
5,367,633	11/1994	Matheny et al.	395/164
5,369,766	11/1994	Nakano et al.	395/700
5,371,891	12/1994	Gray et al.	395/700
5,418,964	5/1995	Lonner et al.	395/700
5,423,841	6/1995	Bunke et al.	395/700
5,437,025	7/1995	Bale et al.	395/600
5,485,671	1/1996	Stutz et al.	395/700
5,515,536	5/1996	Corbett et al.	395/700
5,619,638	4/1997	Duggan et al.	395/703

OTHER PUBLICATIONS

Huy Hguyen et al., *OODDM: An Object–Oriented Database Design Model*, Apr., 1990, see page 339, right–hand column, line 35; and page 341, left–hand column, line 2.

C. Popien et al., *A Concept For An ODP Service Management*, IEEE, 1994, see pages 888–897.

Cheng et al, "On The Performance Issues of Object–Based Buffering", Proceedings of the First International Conference on Parallel and Distributed Information Systems, p. 30–37, 4 Dec. 1991.

Primary Examiner—James P. Trammell

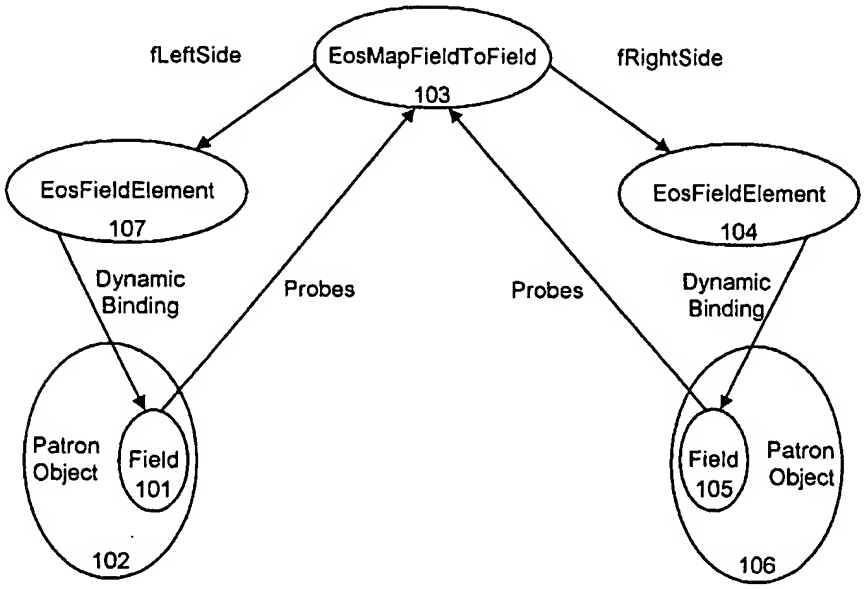
Assistant Examiner—Kakali Chaki

Attorney, Agent, or Firm—Wilson Sonsini Goodrich & Rosati

[57] ABSTRACT

A method and system for creating named relations between classes in a dynamic object-oriented programming environment via mappers is disclosed. The mapping objects dynamically bind to the class interfaces of the classes being related. These connections between classes are defined within a visual environment. The relationships can be programmatically attached by name to object instances during program execution. Because these relationships are stored in a resource and are dynamically bound by name to the objects, they can be created and modified without requiring the source code of the objects being associated to be changed. This eliminates hard coded dependencies between objects that impede reuse of the objects in other contexts. The invention requires and takes full advantage of, metadata, full dynamic binding and probing support in the objects being connected with the invention.

33 Claims, 4 Drawing Sheets



-continued

```
EosAtom &name,
        EosData &data),
  // sender is EosConnection::rootObjectChanged
  virtual void rootObjectChanged(EosObjectRef &newRoot);
  // sender is EosConnection::dataChanged
  void dataChanged(const EosData &data);
  void probeObjectChanged(EosProbeObject &po);
virtual EosSDO *eosGetSDO() const {return NULL;}
protected:
        long
fHandleToSelf;
        long
fHandleToClient;
        EosProbedObjectPath fPath;
        EosObject              *fRoot;
        EosConnection          *fConnection;
        EosProbeID              fProbeId;
        EosProbeObject         *fObjectBeingProbed;
        EosBoolean              fInDataChanged;
};
```

What is claimed is:

1. A system for dynamically linking a first object instance and a second object instance written using a dynamic object-oriented language, comprising at least one semantic link relating said first and said second objects instances through their class interfaces using the dynamic binding capabilities of said first and second objects wherein the semantic link comprises:

a third object instance that sets at least one probe on a field of the object instance, the probe causing the third object instance to perform an action on the second object instance if the field changes and

properties controlling the behavior of the third object instance, the properties

not being changed at runtime and

being stored separately from program code.

2. The system of claim 1 wherein said first and second objects are the same object instance.

3. The system of claim 1 further comprising a plurality of semantic links and means for creating a connection comprising a named list of said plurality of links.

4. The system of claim 3 further comprising a resource for storing a specification of said connection.

5. The system of claim 4 further comprising means for creating and initializing said connection during program execution in accordance with said specification.

6. The system of claim 3 further comprising means for dynamically creating and destroying said connection during program execution.

7. The system of claim 1 wherein said link performs transformations on information passed between said first and said second objects.

8. The system of claim 1, wherein the first object instance resides on a first machine and the second object instance resides on a second machine, the first machine and the second machine coupled to a computer network.

9. The system of claim 1, wherein the third object instance resides on a third machine, the third machine coupled to the computer network.

10. The system of claim 1, wherein the action comprises setting a field in the second object instance.

11. The system of claim 1, wherein the action comprises calling a function in the second object instance.

12. The system of claim 11, wherein the action comprises setting a field in the second object instance and calling a function in the second object instance.

13. The system of claim 1, wherein the action comprises setting a field in the second object instance and calling a function in the second object instance.

14. The system of claim 1, wherein the third object instance sets a second probe on a field of the second object instance, the second probe causing the third object instance to perform an action on the first object instance if the field of the second object instance is changed.

15. A method of establishing communication between a first object instance and a second object instance, the first object and the second object written using a dynamic object-oriented language, the method comprising:

instantiating a third object instance, the third object instance having properties that

are not changed at runtime and

are stored separately from program code;

the third object instance setting a first probe on the first object; and

based on the probe, if a field changes in the first object instance, communicating with the second object instance.

16. The method of claim 15, wherein communicating with the second object instance comprises:

calling a function in the second object instance.

17. The method of claim 16, wherein calling a function comprises:

the third object instance collecting parameters from the first object via dynamic binding and

the third object instance passing the parameters to the function.

18. The method of claim 15, wherein communicating with the second object instance comprises:

updating a field in the second object instance.

19. The method of claim 15, wherein communicating with the second object instance comprises:

firing a second probe on a field of the second object instance.

20. The method of claim 15, comprising

the third object instance setting a second probe on the second object instance and

based on the second probe, if a field changes in the second object instance, communicating with the first object instance.

21. The method of claim 15, comprising:

instantiating a plurality of objects instances;

the object instances in the plurality of object instances setting probes on the first object instance; and

based on at least a probe in the plurality of probes, if a field changes in the first object instance, communicating with the second object instance.

22. The method of claim 15, comprising:

the third object instance transforming information passed between the first object instance and the second object instance.

23. The method of claim 15, wherein the first object instance resides on a first machine and the second object instance resides on a second machine, the first machine and the second machine coupled to a computer network.

24. The method of claim 23, wherein the third object instance resides on a third machine, the third machine coupled to the computer network.

25. A method of dynamically linking a first object instance and a second object instance written using a dynamic object-oriented language, the method comprising;

instantiating a third object instance, the third object instance having properties controlling the behavior of the third object instance, the properties being stored separately from program code;

the third object instance setting a probe on a field of said first object instance, the third object instance setting the probe using a class interface of the first object instance; and

the probe causing the third object instance to perform an action an the second object instance if the field changes.

26. The system of claim **25**, wherein the action comprises setting a field in the second object instance.

27. The system of claim **25**, wherein the action comprises calling a function in the second object instance.

28. The method of claim **27**, wherein calling a function comprises:

the third object instance collecting parameters from the first object instance via dynamic binding and

the third object instance passing the parameters to the function.

29. The method of claim **25**, comprising

the third object instance setting a second probe on a field of the second object instance, the second probe causing the third object instance to perform an action on the first object instance if the field of the second object instance is changed.

30. The method of claim **25**, comprising:

instantiating a plurality of object instances;

the object instances in the plurality of object instances setting a plurality of probes on fields in object instances from among the first object instance and the second object instance, and

probes from among the plurality of probes performing actions on object instances from among the first object instance and the second object instance if respective fields are changed.

31. The method of claim **25**, comprising:

the third object instance transforming information passed between one of said first and second object instances and another of said first and second object instances.

32. The method of claim **25**, wherein the first object instance resides on a first machine and the second object instance resides on a second machine, the first machine and the second machine coupled to a computer network.

33. The method of claim **32**, wherein the third object instance resides on a third machine, the third machine coupled to the computer network.

* * * * *

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 9/46

A1

(11) International Publication Number: **WO 97/15883**

(43) International Publication Date: 1 May 1997 (01.05.97)

(21) International Application Number: PCT/US96/16927

(22) International Filing Date: 22 October 1996 (22.10.96)

(30) Priority Data:
08/548,536 26 October 1995 (26.10.95) US

(71) Applicant: VIEWSOFT, INC. [US/US]; Suite 300, 250 West Center Street, Provo, UT 84601 (US).

(72) Inventors: MITCHELL, David, C.; 1823 South 250, South Orem, UT 84058 (US). ANDERSON, Kelly, L.; 1962 East California Avenue, Provo, UT 84606 (US). OSMAN, Andrew, V.; 231 South 100 East, Provo, UT 84606 (US). MITCHELL, Dale, K.; 361 North 300 West #303, Provo, UT 84601 (US).

(74) Agent: CHAN, Hark, C.; Wilson, Sonsini, Goodrich & Rosati, 650 Page Mill Road, Palo Alto, CA 94304-1050 (US).

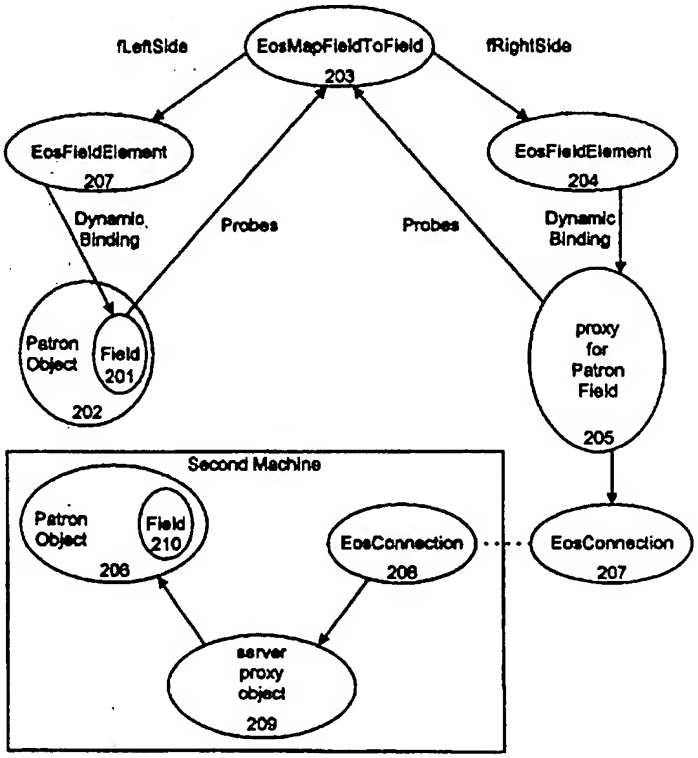
(81) Designated States: JP, European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: METHOD FOR MANAGING DYNAMIC RELATIONS BETWEEN OBJECTS IN DYNAMIC OBJECT-ORIENTED LANGUAGES

(57) Abstract

A method and system for creating named relations between classes in a dynamic object-oriented programming environment via mappers is disclosed. The mapping objects dynamically bind to the class interfaces of the classes being related. These connections between classes are defined within a visual environment. The relationships can be programmatically attached by name to object instances during program execution. Because these relationships are stored in a resource and are dynamically bound by name to the objects, they can be created and modified without requiring the source code of the objects being associated to be changed. This eliminates hard coded dependencies between objects that impede reuse of the objects in other contexts. The invention requires and takes full advantage of, meta-data, full dynamic binding and probing support in the objects being connected with the invention.



WHAT IS CLAIMED IS:

1. A system for dynamically linking a first object instance and a second object instance written using a dynamic object-oriented language, comprising at least one semantic link relating said first and said second objects through their class interfaces using the dynamic binding and probing capabilities of said first and second objects.

2. The system of claim 1 wherein said first and second objects are the same object instance.

3. The system of claim 1 further comprising a plurality of semantic links and means for creating a connection comprising a named list of said plurality of links

4. The system of claim 3 further comprising a resource for storing a specification of said connection.

5. The system of claim 4 further comprising means for creating and initializing said connection during program execution in accordance with said specification.

6. The system of claim 3 further comprising a third object and means for maintaining said connection during program execution by changing said one link to relate said first and said third objects when a characteristic of one of said first and said second objects changes.

7. The system of claim 3 further comprising means for destroying said connection during program execution.

8. The system of claim 1 wherein said link performs transformations on information passed between said first and said second objects.

58/3,K/115 (Item 115 from file: 348) **Links**
EUROPEAN PATENTS

00593835
Computer simulation system and method for specifying the behavior of graphical operator interfaces.
Rechnersimulationssystem und Verfahren zum Spezifizieren vom Verhalten graphischer Bedienerschnittstellen.
Systeme de simulation par ordinateur et methode pour specifier le comportement d'interfaces d'operateur graphiques.
PATENT ASSIGNEE:
VIRTUAL PROTOTYPES, INC., (1707660), 5252 De Maisonneuve West, Suite 318, Montreal, Quebec H4A 3S5, (CA), (applicant designated states: AT;CH;DE;FR;GB;IT;LI;NL;SE)
INVENTOR:
Joseph,Eugene R., 4692 Roslyn Street, Montreal(Quebec), (CA)
Trachtman,Michael, 542 California Street, Newton,Massachusetts 02160, (US)
LEGAL REPRESENTATIVE:
Zeitler & Dickel (101201), Patentanwalte European Patent Attorneys Herrnstrasse 15, D-80539 Munchen, (DE)
PATENT (CC, No, Kind, Date): EP 597316 A2 940518 (Basic)
EP 597316 A3 950816
APPLICATION (CC, No, Date): EP 93117333 931026;
PRIORITY (CC, No, Date): US 972779 921109
DESIGNATED STATES: AT; CH; DE; FR; GB; IT; LI; NL; SE
INTERNATIONAL PATENT CLASS (V7): G06F-009/44;
ABSTRACT WORD COUNT: 137

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPABF2	1826
SPEC A	(English)	EPABF2	12012
Total word count - document A			13838
Total word count - document B			0
Total word count - documents A + B			13838

INTERNATIONAL PATENT CLASS (V7): G06F-009/44

...SPECIFICATION of the object light XYZ, and extracting the event "Turn OFF" from a pop-up menus. Actions impacting virtual objects are specified in a similar manner.

In conjunction with the prior prototyping systems described, which provide a graphical design environment for virtual object appearance, virtual object behavior and virtual object interfaces, the invention closes the loop for allowing full pictorial programming in the domain of dynamic, graphical operator interfaces. Pictorial programming allows computer users who do not have a traditional programming background to carry...

...The invention is further directed to a visual method of, and system for, programming interactive, graphical applications on a computer system which includes a graphical display and input devices, such as a mouse and

US EQUIV. ATTACHED

a keyboard. The method is particularly well...

...graphical output which changes to reflect the values of data represented by virtual objects.

The visual application is made up of virtual objects, which have both a graphical representation and a behavioral model. The graphical representation of a virtual object is defines using a graphics editor, or otherwise imported into the prototype system. A behavioral model is specified by a Property Sheet that maps a behavioral model for a virtual object to the visual representation. The mapping is specified by pointing to features of the virtual object, such as line of motion or center of rotation, or typing-in values.

The behavioral...

...graphical representations to simulate a physical light.

Virtual objects are connected to, or associated with, data values so as to animate the objects in response to the data. As the data changes, the virtual objects are redrawn to reflect the changed data value. This data can be generated by physical input devices, other virtual objects, software programs residing on the same or another computer, etc.

The subject computer prototype development...

...the user to specify the reaction to events in a pictorial manner, in a visual object environment. For this purpose, a spreadsheet like State Table and the visual object collection coincide on the same graphical display. The State Table is filled in by pointing to lists of virtual events or actions associated with the different virtual objects. The contents of these lists are dependent on the virtual object class. Event or action descriptions are entered into the respective cells of the State Table, in the form of descriptive strings of text. This text describes the events or actions, and the event sources, or action destinations.

The State Table allows the user to specify prototype state names, logical conditions, and the name of the...

...user interface. This information is supplied by a user employing an alphanumeric keyboard, selecting from lists, or copying parameters from other cells of the spreadsheet State Table. Other control paradigms can be implemented using the same method, such as FSM, Rule based systems, or Petri Nets. The data stored in the State Table is used to automatically derive the control program for the prototype.

Furthermore, in conjunction with the information stored in the virtual objects, a complete source code program can be generated. The source code program, when compiled, linked...

...computer system as on the original prototype development platform.

The invention includes a State Table Editor as an interactive module for specifying the behavior of a prototype using, for example, the commercially available Virtual Applications Prototyping System (VAPS). The State Table Editor or definition module interface is similar to a spreadsheet paradigm, in that the behavior of...

...Table Editor replaces the Logic Editor of VAPS to define relationships between and among virtual objects, events, states and actions. The State Table Editor is compatible with various simulation and prototyping systems. Thus, Augmented Transition Network (ATN) language programs...

...and simulation system. In addition, ATN programs can be converted into State Tables and State Tables can be converted into ATN programs.

The State Table Editor allows the user to specify the behavior of a simulator prototype using a spreadsheet like...

...how the prototype should respond to a specified event.

In its simplest form, a State Table specification for a prototype consists of a collection of Reaction Rules. This collection of rules is called a Reaction Table. More sophisticated prototypes can contain more than one rule set. Each rule set is ...complete the State Table by clicking on the prototype elements, i.e., designating a virtual object by manipulation of a graphics cursor using a graphic input device such as a mouse. Then, using...

...the desired behavior of the virtual object.

The State Table Editor package includes context sensitive menus and windows to enable an inexperienced user to define prototype behavior. It allows the expert...

...C" or ADA.

For example, if the user is specifying the behavior of a virtual object , then the State Table editor displays or "pops-up" a window which enumerates all the functions which can be performed by that virtual object.

If the virtual object is an input object, then all the input events which that virtual object can produce is shown in a menu. If the virtual object is an output object, then all the functions which can control the virtual object is displayed. Once the appropriate event or function is chosen, a second pop-up window allows the user to fill in the details/parameters of the virtual object. Default values are nominally provided. If the parameter can have only a predefined set of allowable values, then these values are also displayed in a pop-up menu.

The State Table Editor converts the sequence of key clicks and typed values into the appropriate textual syntax. The textual...

...provides a front end to the ATN capability of the prototype development system. The State Table defined by the user is translated into an ATN program which is compiled and used to drive the target prototype system at runtime.

The State Table Editor allows a user who has little or no programming experience to specify the desired behavior of a prototype by using a mouse to point to the virtual objects which are part of the behavior, and by choosing the desired behavior from context sensitive menus.

US005485600A

United States Patent [19]

Joseph et al.

[11] Patent Number: 5,485,600

[45] Date of Patent: Jan. 16, 1996

[54] **COMPUTER MODELLING SYSTEM AND METHOD FOR SPECIFYING THE BEHAVIOR OF GRAPHICAL OPERATOR INTERFACES**

[75] Inventors: **Eugene R. Joseph**, Montreal, Canada; **Michael Trachtman**, Eugene, Oreg.

[73] Assignee: **Virtual Prototypes, Inc.**, Montreal, Canada

[21] Appl. No.: **972,779**

[22] Filed: **Nov. 9, 1992**

[51] **Int. Cl.**[6] **G06F 17/50**

[52] **U.S. Cl.** **395/500**; 395/161; 364/190; 364/191

[58] **Field of Search** 395/500, 159, 395/161; 364/578, 188, 189, 190, 191, 192, 146, 147, 474.22, 474.23, 474.24, 474.26, 474.27

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,315,315	2/1982	Kossiakoff	395/700
4,849,880	7/1989	Bhaskar et al.	
4,860,204	8/1989	Gendron et al.	395/700
4,868,785	9/1989	Jordan et al.	395/140
4,870,561	9/1989	Love et al.	395/500
4,914,568	4/1990	Kodosky et al.	364/578
5,251,159	10/1993	Rowson	395/500
5,255,363	10/1993	Seyler	395/149
5,261,043	11/1993	Wolber et al.	395/159
5,297,250	3/1994	Leroy et al.	395/157
5,325,481	6/1994	Hunt	395/157

OTHER PUBLICATIONS

"Vaps by Example," Virtual Prototypes Inc., 1991.

"Vaps Installation Guide and Options," Virtual Prototypes Inc. 1991.

"Vaps Conceptual Overview," Virtual Prototypes Inc., 1991.

"Vaps Reference Guide," Virtual Prototypes Inc., 1991.

"Vaps Programmer's Guide," Virtual Prototypes Inc., 1991.

"Vaps Overview," Virtual Prototypes Inc., 1993.

"Vaps with stateforms Reference Guide," Virtual Prototypes Inc., 1993.

Research Disclosure No. 317, Sep. 1990, Emsworth, GB, p. 749 Disclosure No. 31761 "Intelligent Object-Oriented Graphics".

IEEE Software, vol. 6, No. 3, May 1989, New York, US, pp. 15–23, Brad A. Myers: "User-Interface Tools: Introduction and Survey".

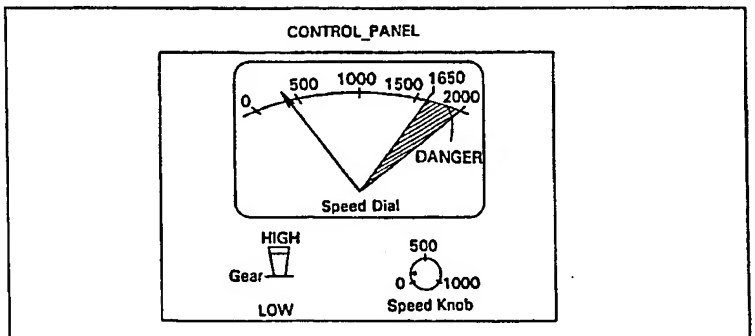
Proceedings of the European Computer Graphics Conference and Exhibition (Eurographics '89), 4 Sep. 1989, Hamburg, FRG, pp. 3–14, Qijing Mao: "Micro-UIDT: A User Interface Development Tool".

Primary Examiner—Parshotam S. Lall
Assistant Examiner—Richard Ellis
Attorney, Agent, or Firm—Lowe, Price, LeBlanc & Becker

[57] **ABSTRACT**

A computer simulator system allows the user to specify prototype reaction to events in a pictorial manner using a visual object environment. A spreadsheet like State Table and a visual object collection coincide on a common graphical display. The State Table is filled in by pointing to lists of events or actions associated with the different objects. The contents of these lists are dependent on the object class. Event or action descriptions are transported into the respective cells of the State Table in the form of descriptive strings of text. This text describes the event or action, and the event source, or action destination. Entries in the State Table define the operation of the simulation and are executed directly by an interpreter or are compiled to generate a program of instructions for performing the simulation.

37 Claims, 23 Drawing Sheets



STATE	EVENT	ACTION	NEXT STATE
LOW	FIRST_TIME	Get_Frame("","CONTROL_PANEL", 1,FOREGROUND_FRAME);	
	INITIALLY	Drive_Dial("","speed_dial",KNOB_VAL);	
	KNOB speed_knob	Drive_Dial("","speed_dial",KNOB_VAL);	
	SWITCH gear 'HIGH'		HIGH
HIGH	INITIALLY	Drive_Dial("","speed_dial",KNOB_VAL*2);	
	KNOB speed_knob	Drive_Dial("","speed_dial",KNOB_VAL*2);	
	SWITCH gear 'LOW'		LOW

As demonstrated, the process of creating a State Table is relatively simple. Most of the required information in a State Table can be entered with mouse clicks. When the default parameters that are entered into the State Table are not correct, the entries can be easily changed. When necessary, the user can click twice on any field, and manually type in or edit the desired text.

In summary, the prototype development system according to the invention provides a system and method for defining the responses and states of a simulated system using a series of context sensitive menus to generate state tables. The information from the state tables is used to generate program code used by the prototype development platform or a target platform to simulate a desired system.

Although the present invention has been described and illustrated in detail, it is clearly understood that the same is by way of illustration and example only and is not to be taken by way of limitation, the spirit and scope of the present invention being limited only by the terms of the appended claims. For example, although the State Table Editor has been described in the context of a system for simulating a physical system, the invention is also applicable to other systems such as real-time control, artificial intelligence, and other data processing systems the operation of which can be defined by a state transition table.

We claim:

1. A system for developing and executing and specifying interactive visual applications wherein dynamic data are mapped into coherently animated virtual objects and operator interaction facilities are provided to interact with the objects on the display, the system comprising:

a central processing unit responsive to operator control signals for deriving address and control signals;

a random access memory for providing instructions to said central processing unit in response to said address signals from said central processing unit;

a non-volatile storage medium for supplying said instructions to said random access memory in response to said control signals from said central processing unit;

a graphics display device responsive to said control signals from said central processing unit for displaying the objects; and

operator input means for supplying said operator control signals to said central processing unit, said operator control signals including positional and alphanumeric components, said operator input means including:

(i) a locator/trigger device for supplying said positional signals to the central processing unit, and

(ii) a data entry device for supplying said alphanumeric signals to said central processing unit;

said instructions including at least one functional module for defining operations of said central processing unit and including:

(i) an object editor module responsive to said operator control signals for deriving the virtual objects by relating a graphical appearance of said virtual objects to a behavior of said objects during execution by the system;

(ii) an integration editor module responsive to said operator control signals for mapping ones of the virtual objects into names of variables, the contents of memory locations of said random access memory corresponding to said named variables, said named variables being used for redrawing the virtual objects on the graphics display to reflect changes in the contents of the memory locations associated with said named variables and said named variables further supplying graphic access to data in response to said positional signals from said locator/trigger device and to screen locations of said virtual objects,

(iii) an execution facility for redrawing ones of said virtual objects in response to at least one of: (i) dynamic data stored in said random access memory and (ii) operator control signals, the execution facility further being responsive to predetermined logic specification data;

(a) said logic specification data specifying sets of events and corresponding reactions which should occur in response to the respective events, a respective origin of the event, a state of the system wherein said state is a state in a finite state machine model of the system, wherein each of said events originating from at least one of (A) ones of said objects, (B) a system clock signal, (C) control signals from said central processing unit, (D) messages from other computer systems, or (E) changes in the contents of the named variables;

(b) said logic specification data further specifying the next state that the system should transition to after an event is received;

(c) said logic specification data further specifying the reactions that occur under control of the logic specification, ones of which affect ones of the objects displayed on the display device; and

a state table editor responsive to said operator control signals for defining said logic specification by specifying:

(1) system states;

(2) events to be recognized in respective ones of said states;

(3) actions to be taken in response to respective ones of said events and states; and

(4) next states to be entered in response to respective ones of said events and states.

2. The system of claim 1 wherein said logic specification is generated responsive to selection of the virtual objects displayed on the graphics display to fill in a state table including said events and conditions to be recognized and corresponding actions to be taken.

3. The system of claim 2 including means responsive to said operator control signals for entering state names of respective ones of said states into the state table.

4. The system of claim 3 wherein said means for entering said state names includes a data entry device.

5. The system of claim 3 wherein said means for entering said state names includes means for copying from one of said cells forming said state table to another of said cells forming said state table.

6. The system of claim 1 including means for specifying events and respective sources of said events by clicking with the locator/trigger device on the visual representation of the virtual object generating respective ones of said events.

7. The system of claim 6 including means responsive to said locator/trigger device for selectively inserting said logic specification data into said state table.

8. The system of claim 1 further comprising means for specifying said actions by clicking with the locator/trigger device on visual representations of the virtual objects to which said actions apply.

9. The system of claim 8 including means for selectively inserting ones of said actions into cells forming the state table.

10. The system of claim 9 wherein said means for selectively inserting comprises said locator/trigger device.

11. The system of claim 1 including non-volatile memory means for storing said logic specification in an open ASCII format and a description of objects aggregates in an open ASCII format, whereby for each object such description includes details of the object class, object interface to a controlling variable and object graphical primitives, attributes and transformations.

12. The system of claim 1 wherein said execution facility includes said logic specification and said execution facility is stored in said random access memory during a system execution mode of operation.

13. The system of claim 1 wherein said logic specification and object description is postprocessed and stored in the modeled system, wherein the modeled system exhibits a behavior substantially the same as a behavior of said model system.

14. The system of claim 13 wherein the logic specification and object description includes states and transition statements, each transition statement including: an event, an action list and a next state, each event including an event type and an event descriptor, an action list being a list of names of programs that are to be executed, and the next state being a name,

the object description including, for any object, behavioral parameters, interface information and graphics information; the behavioral information being class dependent and capturing salient real-time behavior of the object class, the interface information identifying at least one variable through which the application interfaces with the operator interface, the graphics information including (i) graphics primitives and coordinates thereof, (ii) graphics transformations such as scaling, rotation, and translation and (iii) graphics attributes applicable to any primitive, such as color, line style, texture and fonts.

15. The system of claim 14 wherein at least some of the transition statements include a condition, the condition being any expression involving memory addresses.

16. The system of claim 1 wherein said instructions further include:

(i) an object editor module responsive to said operator control signals for defining the virtual objects by relating a graphical appearance of said virtual objects to a behavior of said objects during execution by the system;

(ii) an integration editor module responsive to said operator control signals for mapping ones of the virtual objects into names of variables, the contents of memory locations of said random access memory corresponding to said named variables, said named variables being used for redrawing the virtual objects on the graphics display to reflect changes in the contents of the memory locations associated with said named variables and said named variables further supplying graphic access to data in response to said positional signals from said locator/trigger device and to screen locations of said virtual objects,

(iii) an execution facility for redrawing ones of said virtual objects in response to dynamic data stored in said random access memory and in response to operator control signals, the execution facility further responsive to predetermined logic specification data.

17. The system of claim 1 including means for the operator interface to communicate with an application by transferring data between the operator interface and the application.

18. The system of claim 1 wherein said logic specification data further specify predetermined conditions existing when the events occurred, and the state table editor further specifies conditions to be recognized in respective ones of said system states specified by the state table editor.

19. The system of claim 1 wherein the object editor module includes:

I. at least one of: (A) a first arrangement for creating or modifying custom graphics including facilities for: (i) precisely drawing graphic primitives, such as lines, circles, polygons and texts; (ii) manipulating at least one of the color, texture, line style and font graphic attributes of said primitives; (iii) combining such primitives to form hierarchical constructions; and (iv) selectively applying graphic transformations, such as scaling, translating, rotating and combinations thereof, to the primitives or hierarchical constructions thereof; and (B) a second arrangement for building custom graphics with another tool and importing custom graphics into the object editor; and

II. A third arrangement for mapping user defined graphics derived by at least one of the first and second arrangements into objects with execution time behaviors, the execution time behaviors being mapped to the user defined graphics by selecting these graphics through pointing and clicking with the graphics input device, and by filling in class specific information into a class specific form by pointing and clicking with the graphics input device on the displayed graphics and using an alphanumeric data entry device, resulting in an association of the execution time behaviors to the imported graphics, the execution time behaviors including at least one of:

a. displaying dynamic data of the application as a user designed bar chart,

b. displaying a finite, but dynamically varying, number of any of the user designed visual objects with execution behaviors specified herein in a clipped viewing area,

c. displaying and controlling a user defined cursor having an arbitrary graphical looks inside a clipped viewing area,

d. displaying dynamic real-time application data in a rotational manner, by rotating user defined or the imported graphics around a user defined center of rotation,

e. displaying as a graphics overlay user defined animated graphics over the graphics produced by a user program,

f. forming a correspondence between a set of user defined graphical representations and a finite set of natural numbers (1, 2, 3, . . . n), the natural numbers dynamically varying under application control and having only one representation of the set displayed at any given time based on the value of an index of the set,

g. displaying dynamic real-time application data in a textual fashion as an ASCII string, using a font,

h. displaying dynamic real-time application data as a user designed graph that is redrawn dynamically to illustrate the most recent data values,

i. displaying dynamic real time application data as a set of user designed graphics translating along a straight line segment,

j. forming mapping between a finite set of consecutive natural numbers and a rectangular collection of graphics and displaying the portion of the rectangular collection of graphics that corresponds to a dynamically computed, current value in the set of numbers inside a rectangular region,

k. sending to the application a value selected dynamically from a finite set of values, while concurrently providing visual feedback by displaying one of a set of user designed graphics mapped to that value, where such value set is traversed in a fixed order,

l. sending a user defined ASCII string to the application, the user defined ASCII string being statically associated with a user defined graphic appearing on the display, the user defined ASCII string being sent to the application in real-time whenever the user dynamically interfaces with the displayed graphics,

m. defining and referencing a collection of graphics to be displayed or removed together under application control,

n. sending to the application a character string generated by typing on a real hardware keyboard or by inputting through a virtual keyboard comprised of user defined graphics appearing on the display, after validation and display of the generated character string using a font, said character string being sent in real-time as the user manipulates the real or virtual keyboard,

o. sending to the application ASCII data associated with user defined graphics appearing in a set of non-overlapping rectangular regions on the display, such that for every region, a distinct character or character string is sent, the ASCII data associated with user defined graphics being sent in real-time as the user manipulates the user defined graphics,

p. sending to the application a continuous input value varying in a finite, continuous real-number range, and concurrently providing real-time visual feedback by rotating a set of user defined graphics around a user defined center of rotation,

q. sending to the application an input signal including x,y coordinates derived by associating a user defined coordinate set with a user defined graphics region and selecting at execution time a point in the graphics region to thus map to the x,y coordinate set,

r. sending to the application lists of ASCII strings contained in pop-up or pull-down menus, said menus including user designed graphics and a user designed look and feel while being manipulated, the strings in the menus being sent to the application in real-time by manipulating the graphics through the graphic input device,

s. sending to the application a continuously varying input value from a finite continuous range while concurrently providing visual feedback by translating user designed graphics along a straight line segment mapped to a value range,

t. causing the application to display graphics primitives and alter graphical attributes thereof such as color, font, texture and transform them, such as by rotating, translating, scaling, making visible and invisible, based on real-time values of an application variable,

u. sending to the application a value from a finite set of discrete values, and providing concurrent visual feedback by rotating an image on the display around a point at predefined angles,

v. forming correspondence between a graphics object including simultaneously displayed user designed and first and second graphic objects, and sending to the application the identity of the second object when the user interfaces with graphics of the first object, such correspondence being of a temporal nature and existing only when the first and second objects are simultaneously displayed,

w. sending to the application a value from a finite set of discrete values and providing concurrent visual feedback by displaying mutually exclusive user designed graphics on the display, the traversal order of the set being determined at execution time by the user,

x. visually displaying and interacting with application data structured as a collection of nodes and connecting arcs (or links), each node and link being capable of being individually addressed and queried, and each node and link being visually displayed on the display as a network employing user designed graphics,

the data sent to the application being generated in response to a human interaction with at least one of the objects or being supplied to at least one of the objects through a simulated event generation mechanism, the application data being sent from the application to objects built with the subject object editor for display,

the application data being the contents of a finite set of named application variables of various types contained in computer memory accessed by the application program, the application data reflecting the latest results of computations and other data manipulations performed by the application program.

20. The system of claim **19** wherein the imported graphics are graphics custom designed by the user.

21. The system of claim **19** wherein the imported graphics are derived from an external editing tool.

22. The system of claim **19** wherein the fonts are designed by the user.

23. The system of claim **19** wherein the fonts are scalable vector fonts.

24. The system of claim **19** wherein the fonts are user designed raster fonts.

25. The system of claim **19** wherein the fonts are system-provided fonts.

26. The system of claim **19** wherein a new correspondence occurs in response to the second object being replaced on the display by another object.

27. The system of claim **19** further including an integration editor facility for visually defining a run-time correspondence between an object and a named application variable such that a change in application data causes a redrawing of the corresponding graphics object in accordance with its behavioral class.

28. The system of claim **27** wherein manipulation of the graphics object causes a change of the corresponding application data.

29. The system of claim **19** wherein the execution time behaviors include building new, more complex objects through a combination of the above behaviors by making an object part of another object.

30. The system of claim **19** wherein the execution time behaviors include manipulating a collection of objects including multiple hierarchical objects as an entity rather than as individual objects.

31. The system of claim **19** wherein the execution time behaviors include dynamically instancing any number of objects and altering the position thereof and number of instances displayed based on application data.

32. The system of claim **19** wherein all of the execution time behaviors a–x are included, as are the execution time behaviors of claims **29–31**.

33. A system for enabling visual applications to be specified, comprising:

a central processing unit responsive to operator control signals for deriving address and control signals;

a random access memory for providing instructions to said central processing unit in response to said address signals from said central processing unit;

a non-volatile storage medium for supplying said instructions to said random access memory in response to said control signals from said central processing unit;

a graphics display device responsive to said control signals from said central processing unit for displaying the objects; and

operator input means for supplying said operator control signals to said central processing unit, said operator control signals including positional and alphanumeric components, said operator input means including:

(i) a locator/trigger device for supplying said positional signals to the central processing unit, and

(ii) a data entry device for supplying said alphanumeric signals to said central processing unit;

said instructions including at least one functional module for defining operations of said central processing unit and including:

an object editor module responsive to said operator control signals for deriving virtual objects by relating a graphical appearance of said virtual objects to a behavior of said objects during execution by the system,

the object editor module being arranged so a user can import graphics; the object editor module assigning execution time behaviors (i) to said imported graphics or (ii) to user designed graphics; the object editor module having:

a drawing capability for supporting creation of custom graphics providing the user with the capabilities of: (i) precisely drawing graphic primitives, such as lines, circles, rectangles, polygons and texts; (ii) manipulating attributes of at least one of color, texture, line style and font; (iii) selectively applying graphic transformations, such as scaling, translation, rotation and combinations thereof, to the primitives; and (iv) combining such primitives to form more complex constructions; the execution time behaviors being mapped to the imported graphics by (i) selecting graphics on the display through pointing and clicking with the graphics input device, and (ii) filling in class specific information by pointing and clicking with the graphics input device on the displayed graphics and using an alphanumeric data entry device, resulting in an association of the execution time behaviors to the imported graphics, the execution time behaviors including at least one of:

a. displaying dynamic data of the application as a user designed bar chart,

b. displaying a finite, but dynamically varying, number of any of the user designed visual objects with execution behaviors specified herein in a clipped viewing area,

c. displaying and controlling a user defined cursor having an arbitrary graphical looks inside a clipped viewing area,

d. displaying dynamic real-time application data in a rotational manner, by rotating user defined or the imported graphics around a user defined center of rotation,

e. displaying as a graphics overlay user defined animated graphics over the graphics produced by a user program,

f. forming a correspondence between a set of user defined graphical representations and a finite set of natural numbers (1, 2, 3, . . . n), the natural numbers dynamically varying under application control and having only one representation of the set displayed at any given time based on the value of an index of the set,

g. displaying dynamic real-time application data in a textual fashion as an ASCII string, using a font,

h. displaying dynamic real-time application data as a user designed graph that is redrawn dynamically to illustrate the most recent data values,

i. displaying dynamic real time application data as a set of user designed graphics translating along a straight line segment,

j. forming mapping between a finite set of consecutive natural numbers and a rectangular collection of graphics and displaying the portion of the rectangular collection of graphics that corresponds to a dynamically computed, current value in the set of numbers inside a rectangular region,

k. sending to the application a value selected dynamically from a finite set of values, while concurrently providing visual feedback by displaying one of a set of user designed graphics mapped to that value, where such value set is traversed in a fixed order,

l. sending a user defined ASCII string to the application, the user defined ASCII string being statically associated with a user defined graphic appearing on the display, the user defined ASCII string being sent to the application in real-time whenever the user dynamically interfaces with the displayed graphics,

m. defining and referencing a collection of graphics to be displayed or removed together under application control,

n. sending to the application a character string generated by typing on a real hardware keyboard or by inputting through a virtual keyboard comprised of user defined graphics appearing on the display, after validation and display of the generated character string using a font, said character string being sent in real-time as the user manipulates the real or virtual keyboard,

o. sending to the application ASCII data associated with user defined graphics appearing in a set of non-overlapping rectangular regions on the display, such that for every region, a distinct character or character string is sent, the ASCII data associated with user defined graphics being sent in real-time as the user manipulates the user defined graphics,

p. sending to the application a continuous input value varying in a finite, continuous real-number range, and concurrently providing real-time visual feedback by rotating a set of user defined graphics around a user defined center of rotation,

q. sending to the application an input signal including x,y coordinates derived by associating a user defined coordinate set with a user defined graphics region and selecting at execution time a point in the graphics region to thus map to the x,y coordinate set,

r. sending to the application lists of ASCII strings contained in pop-up or pull-down menus, said menus including user designed graphics and a user designed look and feel while being manipulated, the strings in the menus being sent to the application in real-time by manipulating the graphics through the graphic input device,

s. sending to the application a continuously varying input value from a finite continuous range while concurrently providing visual feedback by translat-

ing user designed graphics along a straight line segment mapped to a value range,

t. causing the application to display graphics primitives and alter graphical attributes thereof such as color, font, texture and transform them, such as by rotating, translating, scaling, making visible and invisible, based on real-time values of an application variable,

u. sending to the application a value from a finite set of discrete values, and providing concurrent visual feedback by rotating an image on the display around a point at predefined angles,

v. forming correspondence between a graphics object including simultaneously displayed user designed and first and second graphic objects, and sending to the application the identity of the second object when the user interfaces with graphics of the first object, such correspondence being of a temporal nature and existing only when the first and second objects are simultaneously displayed,

w. sending to the application a value from a finite set of discrete values and providing concurrent visual feedback by displaying mutually exclusive user designed graphics on the display, the traversal order of the set being determined at execution time by the user,

x. visually displaying and interacting with application data structured as a collection of nodes and connecting arcs (or links), each node and link being capable of being individually addressed and queried, and each node and link being visually displayed on the display as a network employing user designed graphics,

the data sent to the application means being generated in response to a human interaction or a simulated event generation mechanism responding to a graphical operator interface built with said modeling system, the application data that is displayed being sent from the application to the graphical operator interface built with the subject modelling system,

the application data being a finite set of named application variables of various types contained in computer memory accessed by the application program, the application data reflecting the latest results of computations and other data manipulations performed by the application program.

34. The system of claim **33** wherein all of the execution time behaviors a–x are included, as are the following execution time behaviors:

building new, more complex objects through a combination of the above behaviors by making an object part of another object;

manipulating a collection of objects including multiple hierarchical objects as an entity rather than as individual objects; and

dynamically initiating any number of objects and altering the position thereof and number of instances displayed based on application data.

35. A modeling, simulation, specification and source code generation system for at least one graphical, data driven operator interface, comprising:

a graphics input device responsive to user inputs for deriving a positional signal;

a processor unit including:

an object editor module responsive to said operator control signals for deriving virtual objects by relating a graphical appearance of said virtual objects to a behavior of said objects during execution by the system, the object editor module being arranged so a user can import graphics, the object editor module assigning execution time behaviors (i) to said imported graphics or (ii) to user designed graphics, the object editor module having:

I. at least one of: (A) a first arrangement for creating and modifying custom graphics including facilities for: (i) precisely drawing graphic primitives, such as lines, circles, polygons and texts; (ii) manipulating at least one of the color, texture, line style, and font graphic attributes of said primitives; (iii) combining such primitives to form hierarchical constructions; and (iv) selectively applying graphic transformations, such as scaling, translating, rotating and combinations thereof, to the primitives or hierarchical constructions thereof and; (B) a second arrangement for building custom graphics with another tool and importing custom graphics into the object editor; and

(II) a third arrangement for mapping user defined graphics derived by at least one of the first and second arrangements into objects with execution time behaviors, the execution time behaviors being mapped to the user defined graphics by selecting these graphics through pointing and clicking with the graphics input device, and by filling in class specific information into a class specific form by pointing and clicking with the graphics input device on the displayed graphics and using an alphanumeric data entry device, resulting in an association of the execution time behaviors to the imported graphics, the execution time behaviors including at least one of:

a. displaying dynamic data of the application as a user designed bar chart,

b. displaying a finite, but dynamically varying, number of any of the user designed visual objects with execution behaviors specified herein in a clipped viewing area,

c. displaying and controlling a user defined cursor having an arbitrary graphical look inside a clipped viewing area,

d. displaying dynamic real-time application data in a rotational manner, by rotating user defined or the imported graphics around a user defined center of rotation,

e. displaying as a graphics overlay user defined animated graphics over the graphics produced by a user program,

f. forming a correspondence between a set of user defined graphical representations and a finite set of natural numbers (1, 2, 3, . . . n), the natural numbers dynamically varying under application control and having only one representation of the set displayed at any given time based on the value of an index of the set,

g. displaying dynamic real-time application data in a textual fashion as an ASCII string, using a font,

h. displaying dynamic real-time application data as a user designed graph that is redrawn dynamically to illustrate the most recent data values,

i. displaying dynamic real time application data as a set of user designed graphics translating along a straight line segment,

j. forming mapping between a finite set of consecutive natural numbers and a rectangular collection of graphics and displaying the portion of the rectangular collection of graphics that corresponds to a dynamically computed, current value in the set of numbers inside a rectangular region,

k. sending to the application a value selected dynamically from a finite set of values, while concurrently providing visual feedback by displaying one of a set of user designed graphics mapped to that value, where such value set is traversed in a fixed order,

l. sending a user defined ASCII string to the application, the user defined ASCII string being statically associated with a user defined graphic appearing on the display, the user defined ASCII string being sent to the application in real-time whenever the user dynamically interfaces with the displayed graphics,

m. defining and referencing a collection of graphics to be displayed or removed together under application control,

n. sending to the application a character string generated by typing on a real hardware keyboard or by inputting through a virtual keyboard comprised of user defined graphics appearing on the display, after validation and display of the generated character string using a font, said character string being sent in real-time as the user manipulates the real or virtual keyboard,

o. sending to the application ASCII data associated with user defined graphics appearing in a set of non-overlapping rectangular regions on the display, such that for every region, a distinct character or character string is sent, the ASCII data associated with user defined graphics being sent in real-time as the user manipulates the user defined graphics,

p. sending to the application a continuous input value varying in a finite, continuous real-number range, and concurrently providing real-time visual feed-back by rotating a set of user defined graphics around a user defined center of rotation,

q. sending to the application an input signal including x,y coordinates derived by associating a user defined coordinate set with a user defined graphics region and selecting at execution time a point in the graphics region to thus map to the x,y coordinate set,

r. sending to the application lists of ASCII strings contained in pop-up or pull-down menus, said menus including user designed graphics and a user designed look and feel while being manipulated, the strings in the menus being sent to the application in real-time by manipulating the graphics through the graphic input device,

s. sending to the application a continuously varying input value from a finite continuous range while concurrently providing visual feedback by translating user designed graphics along a straight line segment mapped to a value range,

t. causing the application to display graphics primitives and alter graphical attributes thereof such as color, font, texture and transform them, such as by rotating, translating, scaling, making visible and invisible, based on real-time values of an application variable,

u. sending to the application a value from a finite set of discrete values, and providing concurrent visual feedback by rotating an image on the display around a point at predefined angles,

v. forming correspondence between a graphics object including simultaneously displayed user designed and first and second graphic objects, and sending to the application the identity of the second object when the user interfaces with graphics of the first object, such correspondence being of a temporal nature and existing only when the first and second objects are simultaneously displayed,

w. sending to the application a value from a finite set of discrete values and providing concurrent visual feedback by displaying mutually exclusive user designed graphics on the display, the traversal order of the set being determined at execution time by the user,

x. visually displaying and interacting with application data structured as a collection of nodes and connecting arcs (or links), each node and link being capable of being individually addressed and queried, and each node and link being visually displayed on the display as a network employing user designed graphics;

the data sent to the application being generated in response to a human interaction with at least one of the objects or being supplied to at least one of the objects through a simulated event generation mechanism, the application data being sent from the application to objects built with the subject object editor for display,

the application data being the contents of a finite set of named application variables of various types contained in computer memory accessed by the application program, the application data reflecting the latest results of computations and other data manipulations performed by the application program;

the processor unit further including: a code generation module for (i) processing object description information including behavioral parameters, interface information and graphics information; the behavioral information being class dependent and capturing the salient real-time behavior of the object class and (ii) generating graphics source code, which after suitable post-processing, in the form of assembly or compilation, link-editing or interpretation, causes an image similar to the one appearing on the model system to appear on the model(ed) system, where the two images have similar animation behaviors when subjected to similar data.

36. The system of claim **35** wherein the code generating module responds to object description information including behavioral parameters, interface information and graphics information to generate graphics source code in any language which can be described formally through a grammar, the behavioral information being class dependent and capturing the salient real-time behavior of the object class, the interface information identifying a variable through which the application interfaces with the operator interface, the graphics information including (i) graphics primitives and coordinates thereof, and (ii) graphics transformations such as scaling rotating, translating, and (iii) graphics attributes applicable to any primitive, such as color, line style, texture and fonts.

37. A modeling, simulation, specification and source code generation system for at least one graphical data driven, interactive operator interface capable of executing similarly on a model system and on a modeled system, comprising:

a graphic input device responsive to user inputs for deriving a positional signal;

a processor unit including:

(1) an input module;

(2) an object editor module responsive to said operator control signals for deriving virtual objects by relating a graphical appearance of said virtual objects to a behavior of said objects during execution by the system, the object editor module being arranged so a user can import graphics, the object editor module

assigning execution time behaviors (i) to said imported graphics or (ii) to a user designed graphics, the object editor module having:

I. at least one of: (A) a first arrangement for creating and modifying custom graphics including facilities for: (i) precisely drawing graphic primitives, such as lines, circles, polygons and texts; (ii) manipulating of at least one of the color, texture, line style, font and texture graphic attributes of said primitives; (iii) combining such primitives to form hierarchical constructions; and (iv) selectively applying graphic transformations, such as scaling, translating, rotating and combinations thereof, to the primitives or hierarchical constructions thereof; and (B) a second arrangement for building custom design graphics with another tool and importing custom design graphics into the object editor; and

(II) a third arrangement for mapping user defined graphics derived by one of (i)–(iv) into objects with execution time behaviors being mapped to the user defined graphics by selecting these graphics through pointing and clicking with the graphics input device, and by filling in class specific information by pointing and clicking with the graphics input device on the displayed graphics and using an alphanumeric data entry device, resulting in an association of the execution time behaviors to the imported graphics, the execution time behaviors including at least one of:

a. displaying dynamic data of the application as a user designed bar chart,

b. displaying a finite, but dynamically varying, number of any of the user designed visual objects with execution behaviors specified herein in a clipped viewing area,

c. displaying and controlling a user defined cursor having an arbitrary graphical look inside a clipped viewing area,

d. displaying dynamic real-time application data in a rotational manner, by rotating user defined or the imported graphics around a user defined center of rotation,

c. displaying as a graphics overlay user defined animated graphics over the graphics produced by a user program,

f. forming a correspondence between a set of user defined graphical representations and a finite set of natural numbers (1, 2, 3, . . . n), the natural numbers dynamically varying under application control and having only one representation of the set displayed at any given time based on the value of an index of the set,

g. displaying dynamic real-time application data in a textual fashion as an ASCII string, using a font,

h. displaying dynamic real-time application data as a user designed graph that is redrawn dynamically to illustrate the most recent data values,

i. displaying dynamic real time application data as a set of user designed graphics translating along a straight line segment,

j. forming mapping between a finite set of consecutive natural numbers and a rectangular collection of graphics and displaying the portion of the rectangular collection of graphics that corresponds to a dynamically computed, current value in the set of numbers inside a rectangular region,

k. sending to the application a value selected dynamically from a finite set of values, while concurrently providing visual feedback by displaying one of a set of user designed graphics mapped to that value, where such value set is traversed in a fixed order,

l. sending a user defined ASCII string to the application, the user defined ASCII string being statically associated with a user defined graphic appearing on the display, the user defined ASCII string being sent to the application in real-time whenever the user dynamically interfaces with the displayed graphics,

m. defining and referencing a collection of graphics to be displayed or removed together under application control,

n. sending to the application a character string generated by typing on a real hardware keyboard or by inputting through a virtual keyboard comprised of user defined graphics appearing on the display, after validation and display of the generated character string using a font, said character string being sent in real-time as the user manipulates the real or virtual keyboard,

o. sending to the application ASCII data associated with user defined graphics appearing in a set of non-overlapping rectangular regions on the display, such that for every region, a distinct character or character string is sent, the ASCII data associated with user defined graphics being sent in real-time as the user manipulates the user defined graphics,

p. sending to the application a continuous input value varying in a finite, continuous real-number range, and concurrently providing real-time visual feedback by rotating a set of user defined graphics around a user defined center of rotation,

q. sending to the application an input signal including x,y coordinates derived by associating a user defined coordinate set with a user defined graphics region and selecting at execution time a point in the graphics region to thus map to the x,y coordinate set,

r. sending to the application lists of ASCII strings contained in pop-up or pull-down menus, said menus including user designed graphics and a user designed look and feel while being manipulated, the strings in the menus being sent to the application in real-time by manipulating the graphics through the graphic input device,

s. sending to the application a continuously varying input value from a finite continuous range while concurrently providing visual feedback by translating user designed graphics along a straight line segment mapped to a value range,

t. causing the application to display graphics primitives and alter graphical attributes thereof such as color, font, texture and transform them, such as by rotating, translating, scaling, making visible and invisible, based on real-time values of an application variable,

u. sending to the application a value from a finite set of discrete values, and providing concurrent visual feedback by rotating an image on the display around a point at predefined angles,

v. forming correspondence between a graphics object including simultaneously displayed user designed and first and second graphic objects, and sending to the application the identity of the second object when the user interfaces with graphics of the first object, such correspondence being of a temporal nature and existing only when the first and second objects are simultaneously displayed,

w. sending to the application a value from a finite set of discrete values and providing concurrent visual feed-

back by displaying mutually exclusive user designed graphics on the display, the traversal order of the set being determined at execution time by the user,

x. visually displaying and interacting with application data structured as a collection of nodes and connecting arcs (or links), each node and link being capable of being individually addressed and queried, and each node and link being visually displayed on the display as a network employing user designed graphics;

the data sent to the application being generated in response to a human interaction with at least one of the objects or being supplied to at least one of the objects through a simulated event generation mechanism, the application data being sent from the application to objects built with the subject object editor for display,

the application data being the contents of a finite set of named application variables of various types contained in computer memory accessed by the application program, the application data reflecting the latest results of computations and other data manipulations performed by the application program;

(3) a state table editor responsive to operator control signals for defining a logic specification by specifying:

(a) system states;

(b) events to be recognized in respective ones of said states;

(c) actions to be taken in response to respective ones of said events and states; and

(d) next states to be entered in response to respective ones of said events and states; and

(4) a code generation module for generating (a) portable graphics code from an object description including behavioral parameters, interface information and graphics information; the behavioral information being class dependent and capturing salient real-time behavior of the object class, the interface information identifying a variable through which the application interfaces with the operator interface, the graphics information including graphics primitives and coordinates thereof, graphics transformations such as scaling, rotating, translating and graphics attributes applicable to any primitive, such as color, line style, texture and fonts, and (b) logic control code from a logic specification including states and transition statements, each transition statement including: an event, an action list and a next state; each event including an event type and an event descriptor, a condition being any expression involving memory addresses, an action list being a list of names of programs that are to be executed, and the next state being a name; which code, after suitable post-processing causes images with similar appearances, animation behaviors, and interactive behaviors to appear on the model(ed) system.

* * * * *

58/3,K/114 (Item 114 from file: 348) **Links**
EUROPEAN PATENTS

00643321
Method and apparatus for an adaptive texture mapping controller
Verfahren und Vorrichtung zur adaptiven Steuerung der Texturabbildung
Methode et dispositif adaptif de controle de topographie de texture
PATENT ASSIGNEE:
SUN MICROSYSTEMS, INC., (1392732), 2550 Garcia Avenue, Mountain View, California 94043-1100, (US), (Proprietor designated states: all)
INVENTOR:
Kamen, Yakov, 19334 Greenwood Drive, Cupertino, California 95014, (US)
Sabada, Uma, 3475 Granada Avenue, Nr. 386, Santa Clara, California 95051, (US)
LEGAL REPRESENTATIVE:
Wombwell, Francis (46021), Potts, Kerr & Co. 15, Hamilton Square, Birkenhead Merseyside L41 6BR, (GB)
PATENT (CC, No, Kind, Date): EP 622747 A2 941102 (Basic)
EP 622747 A3 950201
EP 622747 B1 000531
APPLICATION (CC, No, Date): EP 94301239 940222;
PRIORITY (CC, No, Date): US 41073 930401
DESIGNATED STATES: DE; FR; IT; NL; SE
INTERNATIONAL PATENT CLASS (V7): G06F-015/10
ABSTRACT WORD COUNT: 286
NOTE:
Figure number on first page: 4

US EQUIV. ATTACHED

LANGUAGE (Publication,Procedural,Application): English; English; English
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	200022	1402
CLAIMS B	(German)	200022	1367
CLAIMS B	(French)	200022	1746
SPEC B	(English)	200022	4824
Total word count - document A			0
Total word count - document B			9339
Total word count - documents A + B			9339

Method and apparatus for an adaptive texture mapping controller
INTERNATIONAL PATENT CLASS (V7): G06F-015/10

...ABSTRACT A2
The present invention provides a method and apparatus for an adaptive texture mapping controller which provides a way for computer graphics system users or other functions in a graphical display system, to trade off object image rendering speed for object image texture quality. This trade-off is accomplished by providing a plurality of control signals to the adaptive texture mapping controller which indicate the level of texture quality that the user or other function desires. Upon recognizing these control signals, the adaptive texture mapping controller selects a computation method to be used in generating pixel values necessary to provide the desired level of image texture quality.

The adaptive texture mapping controller is able to determine an appropriate method for calculating the end points of span sections of scan lines to be used for the display, based upon a function of the knot parameters which correspond to the vertices which describe each edge of a polygon section of the projected object and one or more of the control signals provided, and then to determine an appropriate...

...functions of pairs of knot parameters or pairs of end points respectively.

The adaptive texture mapping controller is able to determine an appropriate computation method for a given one of a...

US005812141A

United States Patent [19]

Kamen et al.

[11] Patent Number: 5,812,141

[45] Date of Patent: Sep. 22, 1998

[54] METHOD AND APPARATUS FOR AN ADAPTIVE TEXTURE MAPPING CONTROLLER

[75] Inventors: Yakov Kamen, Cupertino; Uma Sabada, Santa Clara, both of Calif.

[73] Assignee: Sun Microsystems, Inc., Palo Alto, Calif.

[21] Appl. No.: 592,285

[22] Filed: Jan. 26, 1996

Related U.S. Application Data

[63] Continuation of Ser. No. 41,073, Apr. 1, 1993, abandoned.

[51] Int. Cl.[6] G06T 11/40

[52] U.S. Cl. 345/430; 345/441

[58] Field of Search 395/130, 125; 345/430, 425, 441

[56] References Cited

U.S. PATENT DOCUMENTS

4,835,532	5/1989	Fant	345/136
4,855,934	8/1989	Robinson	395/130
5,097,427	3/1992	Lathrop et al.	395/130
5,109,481	4/1992	Lathrop et al.	395/133
5,179,638	1/1993	Dawson et al.	395/125
5,214,753	5/1993	Lee et al.	395/125
5,253,339	10/1993	Wells et al.	395/126
5,255,352	10/1993	Falk	395/125
5,307,450	4/1994	Grossman	395/123
5,345,541	9/1994	Kelley et al.	395/126

FOREIGN PATENT DOCUMENTS

A-0 550 244 A2 12/1992 European Pat. Off. .

OTHER PUBLICATIONS

Wolberg, George; *Digital Image Warping;* 1990; pp. 189–204.

"Interpolation for Polygon Texture Mapping & Shading", Paul S. Heckbert, et al. Dept. of Electrical Engineering & Computer Science, University of California Berkeley, pp. 1–11, 1991.

Grand Prix, Supplement Technique, 1991, Tetbury UK, pp. 1–20.

Computer Graphics Principles and Practice by J. Foley et al., 1990, pp. 741–747, 806–807 and 820–826.

Survey of Texture Mapping by P. Heckbert, IEEE CG&A 1986, pp. 56–67.

Texture and Reflection in Computer Generated Images by J. Blinn et al., Communications of the ACM, Oct. 1976, v. 19, No. 10, pp. 542–547.

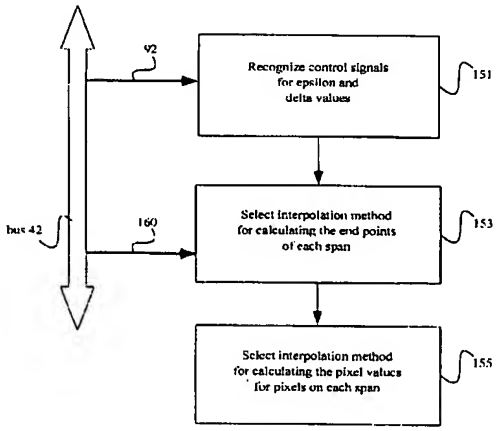
Primary Examiner—Mark K. Zimmerman

Attorney, Agent, or Firm—Beyer & Weaver, LLP

[57] ABSTRACT

The present invention provides a method and apparatus for computer graphics system users or other functions in a graphical display system, to trade off object image rendering speed for object image texture quality. This trade-off is accomplished by providing control signals to the adaptive texture mapping controller which indicate the level of texture quality that the user or other function desires. Upon recognizing these control signals, the adaptive texture mapping controller selects a computation method to be used in generating pixel values necessary to provide the desired level of image texture quality. An appropriate method is determined for calculating the end points of span sections of scan lines to be used for the display, based upon a function of the knot parameters which correspond to the vertices which describe each edge of a polygon section of the projected object and one or more of the control signals provided, and then an appropriate interpolation method is determined for calculating the pixel values for pixels on each span chosen. The selected interpolation method is based on a function of the span end points and one or more of the control signals provided. These functions of the distance between knot parameters on a polygon edge or the distance between end points of a span can be mathematical functions of pairs of knot parameters or pairs of end points respectively.

21 Claims, 8 Drawing Sheets



eters of every internal point (every pixel) on the span are set to the average of the span end point parameters **324**. In the preferred embodiment, the "average" is the arithmetic mean. If "$\delta_0 \leqq MAX\ \{dist_{span}\} \leqq \delta_1$" **326** then the parameters of every internal point (every pixel) on the span are set to a linear combination of the span end point values **328**. In the preferred embodiment, the "linear combination of the span end point values" is calculated as follows:

$T(i)=T(i-1)+\alpha*DELTA,$

$i=0, 1, \ldots, n-1;$

where

T(0)—is a span's start point parameters;

T(n−1)—is a span's end point parameters;

T(i)—is the parameter of the ith-internal point of the span;

α—is a constant coefficient is equal 1/n;

n—is the number of the points in the span;

DELTA—is equal to T(n−1)-T(0).

If "$MAX\ \{dist_{span}\} > \delta_1$" **330** then the parameters of every internal point (every pixel) on the span are computed using the best parameter algorithm available **332**. In this case, the "best parameter algorithm available" is as follows:

$T(i)=NUM(i)/DENOM\ (i),$

$NUM(i)=NUM(i-1)+\alpha*DELTA\_NUM,$

$DENOM(i)=DENOM(i-1)+\alpha*DELTA\_DENOM,$

$i=0, 1, \ldots, n-1;$

where

T(0)—is a span's start point parameter,

T(n−1)—is a span's end point parameter,

T(i)—is the parameter of the ith-internal point of the current span;

α—is a constant coefficient is equal 1/n;

n—is an number of the points in the current span;

NUM(i)—is a numerator of the ith-internal point of the current span;

NUM(0)—is equal T(0)/W(0);

DENOM(i)—is a denominator of the i-internal point of the current span;

DENOM(0)—is equal T(0)/W(0);

DELTA_NUM—is equal T(n−1)/W(n−1)-T(0)/W(0);

DELTA_DENOM—is equal 1/W(n−1)-I/W(0).

Those skilled in these arts will realize that currently, most hardware algorithms use a linear interpolation along each primitive edge to find the parameters for the first and last points of each span. This would correlate to control signal values in the described example of the present invention of $\xi_0=0$, and $\xi_1=\infty$. To achieve the highest quality (but slowest performance), control signal values of $\xi_0=\xi_1=0$, and $\delta_0=\delta_1=0$ would be chosen. Intermediate values for the epsilon and delta control signals will dynamically modify the texture quality/rendering performance scenario, while giving the user more control of the texturing process than previously available in hard coded algorithms. It will be recognized that extensions of this model could include multi-knot parameter computations, such as higher order polynomials and non-linear interpolation.

While the invention has been described in conjunction with a preferred embodiment, it is evident that numerous alternatives, modifications and variations and uses will be evident to those skilled in the art in light of the foregoing description.

What is claimed is:

1. In a computer graphics system for generating and displaying images of objects on a display device, the images including pixels having pixel values, the images being defined by signals representative of the pixel values, the system including elements for processing the signals to modify the images displayed on the display device including, interpolation elements for generating pixel values representative of bitmap image values, which are representative of texture of a given object, the improvement comprising an adaptive texture mapping controller which comprises:

a first mechanism configured to recognize a control signal representing a desired quality of texture for an image of an object, said desired quality of texture ranging from a minimum to a maximum amount of texture provided by said graphics system, said control signal including a plurality of image control values that are selectable by a user, said plurality of image control values being useful in selecting a desired object image rendering speed and image texture quality;

one or more linear and one or more non-linear interpolation mechanisms coupled to said first mechanism for selecting pixels;

a plurality of computation mechanisms couples to said first mechanism for computing values representing image parameters for the selected pixels;

a plurality of hierarchical levels of image parameter computation in which each said hierarchical level requires values representing image parameters to be calculated with a particular interpolation mechanism; and

a computation method selection device, coupled to said first mechanism and using said control signal for selecting one of said interpolation mechanisms and one of said plurality of computation mechanisms to be used in generating pixel values necessary to provide said desired quality of texture, said computation method selection device including logic mechanisms that use selected ones of said plurality of image control values for selecting one of said interpolation mechanisms for calculating first hierarchical level values representing image parameters for a first hierarchical level of image parameter computation, said computation method selection device also including logic mechanisms that use selected ones of said plurality of image control values for selecting one of said interpolation mechanisms for calculating second hierarchical level values representing image parameters for a second hierarchical level of image parameter computation based upon said first hierarchical level values,

whereby a user can trade-off object image rendering speed for object image texture quality by providing said control signal to the adaptive texture mapping controller.

2. The adaptive texture mapping controller of claim **1** wherein the logic mechanisms for selecting one of said interpolation mechanism for calculating first hierarchical level values representing image parameters for a first hierarchical level of image parameter computation further comprises,

a logic mechanism for selecting a computation mechanism for calculating values representing image parameters for end points of a span of a scan line.

3. The adaptive texture mapping controller of claim **2** wherein said logic mechanism for selecting a computation mechanism for calculating values representing image parameters for end points of a span of a scan line, bases said selection of a computation mechanism upon a relation

between a maximum distance between knot parameters and said control signal.

4. The adaptive texture mapping controller of claim 3 wherein said maximum distance between knot parameters can be a mathematical function of values representing image parameters of said knots.

5. The adaptive texture mapping controller of claim 1 wherein the logic mechanisms for selecting one of said interpolation mechanism for calculating second hierarchical level values representing image parameters for a second hierarchical level of image parameter computation further comprises,

a logic mechanism for selecting a computation mechanism for calculating values representing image parameters for selected pixels on a span of a scan line.

6. The adaptive texture mapping controller of claim 5 wherein the logic mechanisms for selecting a computation mechanism for calculating values representing image parameters for selected pixels on a span of a scan line, bases said selection of a computation mechanism upon a relation between a maximum distance between end points of said span and said control signal.

7. The adaptive texture mapping manager of claim 6 wherein said maximum distance between end points of said span can be a mathematical function of values representing image parameters of said end points.

8. In a computer graphics system for generating and displaying images of objects on a display device, the images including pixels having pixel values, the images being defined by signals representative of the pixel values, the system including elements for processing the signals to modify the images displayed on the display device including, interpolation elements for generating pixel values representative of bitmap image values, which are representative of texture of a given object, and a plurality of hierarchical levels in which each said hierarchical level requires values representing image parameters to be calculated with a particular interpolation mechanism, a method for variable control of texture quality for display on a given graphics object, said method comprising the steps of:

providing an adaptive texture mapping controller;

providing one or more linear and one or more non-linear interpolation methods coupled to said adaptive texture mapping controller for computing values representing image parameters; recognizing by said adaptive texture mapping controller, a control signal representing a desired quality of texture for an image of an object, said desired quality of texture ranging from minimum texture to a maximum amount of texture provided by said graphics system, said control signal includes a plurality of image control values selectable by a user in order to select a desired object image rendering speed and image texture quality;

using selected ones of said plurality of image control values included in said control signal for selecting one of said interpolation methods to be used in generating pixel values necessary to provide said desired quality of texture, said selected interpolation method calculates values representing first hierarchical level parameter values for a first hierarchical level of image parameter computation: and

using selected ones of said plurality of image control values included in said control signal for selecting one of said interpolation methods to be used in generating pixel values necessary to provide said desired quality of texture, said selected interpolation method calculates values representing second hierarchical level parameter values for a second hierarchical level of image parameter computation based upon said first hierarchical level parameter values,

whereby a user can trade-off object image rendering speed for object image texture quality by providing said control signal to the adaptive texture mapping controller.

9. The method of claim 8 wherein the step of selecting one of said interpolation method for calculating values representing first hierarchical level parameter values for said first hierarchical level of image parameter computation further comprises a step of selecting an interpolation method for calculating values representing parameter values for end points of a span of a scan line.

10. The method of claim 9 wherein the step of selecting an interpolation method for calculating values representing image parameters for end points of a span of a scan line, bases said selection of an interpolation method upon a relation between a maximum distance between knot parameters and said control signal.

11. The method of claim 10 wherein said maximum distance between values representing knot parameters can be a mathematical function of said values representing knot parameters.

12. The method of claim 11 wherein the step of selecting one of said interpolation method for calculating values representing second hierarchical level parameter values for a second hierarchical level of image parameter computation further comprises a step of selecting one of said interpolation method for calculating values representing parameter values for selected pixels on a span of a scan line.

13. The method of claim 12 wherein the step of selecting an interpolation method for calculating values representing parameter values for selected pixels on a span of a scan line, bases a selection of an interpolation method upon a relation between a maximum distance between end points of said span and said control signal.

14. The method of claim 13 wherein said maximum distance between end points of said span can be a mathematical function of values representing image parameters of said end points.

15. A computer program product embodied on a computer usable medium for controlling speed and quality of texture mapping rendered by a computer graphics system for generating and displaying images of objects on a display device, the images including pixels having pixel values, the images being defined by signals representative of the pixel values, the system including elements for processing the signals to modify the images displayed on the display device including, interpolation elements for generating pixel values representative of bitmap image values, which are representative of texture of a given object, said computer program product comprising:

an adaptive texture mapping controller,

one or more linear and one or more non-linear interpolation mechanisms coupled to said adaptive texture mapping controller for selecting pixels;

a plurality of computation mechanisms coupled to said adaptive texture mapping controller for computing values representing image parameters for selected pixels;

a plurality of hierarchical levels in which each said hierarchical level requires values representing image parameters to be calculated with a particular interpolation mechanism;

a first mechanism configured to permit said adaptive texture mapping controller to recognize a control signal

representing a desired quality of texture which is to be mapped to an object image, said desired quality of texture ranging from minimum to a maximum amount of texture provided by said graphics system, said control signal includes a plurality of image control values selectable by a user, said plurality of image control values being useful in selecting a desired object image rendering speed and image texture quality, said first mechanism includes a third mechanism configured to permit the selection of an interpolation mechanism for calculating first hierarchical level values representing parameter values for a first hierarchical level of image parameter computation; and

a second mechanism configured to permit said adaptive texture mapping controller to use said control signal for selecting one of said interpolation mechanisms and one of said plurality of computation mechanisms to be used in generating pixel values necessary to provide said desired quality of texture, said second mechanism includes a fifth mechanism configured to select an interpolation mechanism for calculating second hierarchical level values representing parameter values for a second hierarchical level of image parameter computation based upon said first hierarchical level values,

whereby a user can trade-off object image rendering speed for object image texture quality by providing said control signal to the adaptive texture mapping controller.

16. The computer program product of claim 15 wherein the third mechanism configured to permit the selection of an interpolation mechanism for calculating values representing first hierarchical level parameter values for a first hierarchical level of image parameter computation further comprises a fourth mechanism configured to select an interpolation mechanism for calculating values representing parameter values for end points of a span of a scan line.

17. The computer program product of claim 16 wherein the fourth mechanism configured to select an interpolation mechanism for calculating values representing parameter values for end points of a span of a scan line, bases a selection of an interpolation mechanisms upon a relation between a maximum distance between values representing knot parameters and said control signal.

18. The computer program product of claim 17 wherein said maximum distance between knot parameters can be a mathematical function of said values representing knot parameters.

19. The computer program product of claim 15 wherein said fifth mechanism configured to select an interpolation mechanism for calculating values representing second hierarchical level parameter values for a second hierarchical level of image parameter computation further comprises a sixth mechanism configured to select an interpolation mechanism for calculating values representing parameter values for selected pixels on a span of a scan line.

20. The computer program product of claim 19 wherein said sixth mechanism configured to select an interpolation mechanism for calculating values representing parameter values for selected pixels on a span of a scan line, bases a selection of an interpolation mechanism upon a relation between a maximum distance between values representing end points of said span and said control signal.

21. The computer program product of claim 20 wherein said maximum distance between values representing end points of said span can be a mathematical function of values representing parameters of said end points.

* * * * *

58/3,K/120 (Item 120 from file: 348)
EUROPEAN PATENTS

00545504
Logical mapping of data objects using data spaces.
Logische Abbildung von Datenobjekten mittels Datenraumen.
Mappage logique d'objets de donnees utilisant des espaces de donnees.
PATENT ASSIGNEE:
International Business Machines Corporation, (200120), Old Orchard Road,
Armonk, N.Y. 10504, (US), (applicant designated states: DE;FR;GB)
INVENTOR:
Fecteau, Jean Gilles, 88 Keewatin Avenue, Toronto, Ontario, (CA)
Kligerman, Eugene, 100 Antibes Drive, Apt 2204, North York, Ontario, (CA)
Kollar, Lubor, 72 Clydesdale Drive, North York, Ontario, (CA)
LEGAL REPRESENTATIVE:
Burt, Roger James, Dr. (52152), IBM United Kingdom Limited Intellectual
Property Department Hursley Park, Winchester Hampshire SO21 2JN, (GB)
PATENT (CC, No, Kind, Date): EP 542483 A1 930519 (Basic)
APPLICATION (CC, No, Date): EP 92310160 921105;
PRIORITY (CC, No, Date): CA 2055295 911112
DESIGNATED STATES: DE; FR; GB
INTERNATIONAL PATENT CLASS (V7): G06F-012/10;
ABSTRACT WORD COUNT: 183

LANGUAGE (Publication,Procedural,Application): English; English; English
Logical mapping of data objects using data spaces.
Mappage logique d'objets de donnees utilisant des espaces de donnees.
INTERNATIONAL PATENT CLASS (V7): G06F-012/10

...ABSTRACT simulating a contiguous data space within a computer memory,
and for placing and accessing data objects of various sizes within the simulated contiguous data space. Multiple, sub-data spaces are concatenated...

...page and each sub-data space in the contiguous data space are uniquely
identified. Data objects are placed in the contiguous data space and at
the first reference to a page...

...the segment containing the referenced page in the contiguous data space
is mapped to the database storage disk. Once a data space page is mapped,
the operating system can read the page into memory without requesting a

US EQUIV.
ATT'D

disk operation from the database manager. On modifying a page, if the database disk page location is changed, the contiguous data space page is
re-mapped without changing the page address in the data space. Also, modified data pages are re-written to the database storage disk in an ongoing manner set by the user, instead of at intervals set...

US005561778A

United States Patent [19]

Fecteau et al.

[11] Patent Number: 5,561,778

[45] Date of Patent: Oct. 1, 1996

[54] SYSTEM FOR REPRESENTING DATA OBJECT IN CONCATENATED MULTIPLE VIRTUAL ADDRESS SPACES WITH COMBINED REQUESTS FOR SEGMENT MAPPING

[75] Inventors: **Jean G. Fecteau**, Toronto; **Eugene Kligerman**; **Lubor Kollar**, both of North York, all of Canada

[73] Assignee: **International Business Machines Corporation**, Armonk, N.Y.

[21] Appl. No.: **442,770**

[22] Filed: **May 17, 1995**

Related U.S. Application Data

[62] Division of Ser. No. 975,245, Nov. 12, 1992, abandoned.

[30] **Foreign Application Priority Data**

Nov. 12, 1991 [CA] Canada 2055295

[51] **Int. Cl.**[6] **G06F 12/08**

[52] **U.S. Cl.** **395/419**; 395/413; 395/416; 395/471; 395/483; 395/876; 395/600; 395/700

[58] **Field of Search** 395/418, 419, 395/416, 471, 483, 600, 700, 876, 413

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,315,310	2/1982	Bayliss et al.	395/823
4,577,274	3/1986	Ho et al.	395/415
4,686,620	8/1987	Ng	395/600
5,301,288	4/1994	Newman et al.	395/412
5,317,731	5/1994	Dias et al.	395/600
5,410,700	4/1995	Fecteau et al.	395/700
5,428,759	6/1995	Smith et al.	395/403

Primary Examiner—Ken S. Kim

Attorney, Agent, or Firm—Sterne, Kessler, Goldstein & Fox P.L.L.C.

[57] **ABSTRACT**

Method and means are provided for simulating a contiguous data space within a computer memory, and for placing and accessing data objects of various sizes within the simulated contiguous data space. Multiple, sub-data spaces are concatenated in such a way that each page and each sub-data space in the contiguous data space are uniquely identified. Data objects are placed in the contiguous data space and at the first reference to a page of the data object, only the segment containing the referenced page in the contiguous data space is mapped to the database storage disk. Once a data space page is mapped, the operating system can read the page into memory without requesting a disk operation from the database manager. On modifying a page, if the database disk page location is changed, the contiguous data space page is remapped without changing the page address in the data space. Also, modified data pages are rewritten to the database storage disk in an ongoing manner set by the user, instead of at intervals set by the operating system.

4 Claims, 16 Drawing Sheets

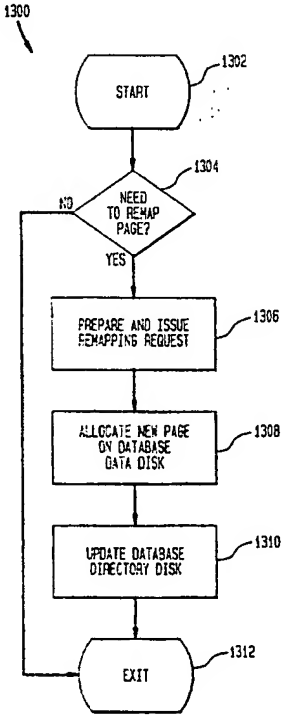


TABLE 3

```
If Data_space_pointer(Data_space_number) = 0 (this data
        space does not already exist) then
do;
    Create the data space
    Set Data_space_pointer(Data_space_number) to the
        sub-data space pointer value used to
        address that data space.
    Allocate the segment valid bit map for that data
        space and initialize it to off.
    Allocate the section_is_modified bit map for that
        data space of size(m/n) and
        initialize it to off.
    Set Modified_group_count(Data_space_number) to 0.
end;
Similarly the page modification procedure can be
represented in pseudocode as shown as follows:
Modify the page.
If Modified_group_count(Data_space_number) > Save_interval
    and Section_is modified(page-address/(n*y)) is on then
do;
    Initialize a save request
    Do i = 0 to (x/n) - 1
        If section_is_modified(i) is on then
            Add pages from (i*n) to ((i*n)+n) to the
            save request
    end;
    Reset Section_is_modified to off
    Issue the save request without waiting for
        completion
end;
If Section_is_modified(page-address/(n*y) is off then
do:  /* none of the pages in this section were
        modified */
    Increment modified_group_count(Data_space_number) by
    1.
    Set Section_is_modified(page-address/(n*y) on.
end;
Similarly the checkpoint procedure can be represented in
pseudocode as shown as follows:
Do j = 1 to max(Data_space_number)
    Initialize a save request
    Do i = 0 to (x/n) - 1
        If section_is_modified(i) for data space(j) is on
            Add pages from (i*n) to ((i*n)+n) to the
            save request
    End;
    Reset Section_is_modified for data space(j) to off
    Issue the save request without wait for completion
End;
Wait for all outstanding save requests to complete.
```

While the invention has been particularly shown and described with reference to a preferred embodiment thereof, it will be understood by those skilled in the art that various other changes in the form and details may be made therein without departing from the spirit and scope of the invention. For example, the various tables mentioned herein may actually be structured as described or may be split into multiple tables to provide for normality of data. Accordingly, the method and system herein disclosed are to be considered merely as illustrative and the invention is to be limited only as specified in the claims.

What is claimed is:

1. A method for mapping on demand a page of a data object contained in a database and stored in one or more database storage disks, wherein the database comprises one or more database data objects, wherein the database is accessed via a contiguous data space representation, the contiguous data space being represented by a plurality of concatenated sub-data spaces, wherein each sub-data space is a virtual data space of a maximum size that is addressable by a computer operating system, each sub-data space comprising a plurality of data segments, each data segment comprising a plurality of pages, wherein the pages in the contiguous data space representation are contiguous, each page comprising a known number of addressable storage locations, wherein the contiguous data space, the sub-data spaces, the data segments, and the pages are addressable by a database management system, wherein the address of a page to be mapped has been determined to be placed in a data segment of a sub-data space, the method comprising the steps of:

(1) determining whether the sub-data space has been created for representation in the contiguous data space;

(2) creating the sub-data space for representation in the contiguous data space if step (1) determines that the sub-data space has not been created in the contiguous data space;

(3) creating a segment bit map for the sub-data space if step (1) determines that the sub-data space has not been created in the contiguous data space, wherein said segment bit map comprises a plurality of bits, each bit representing a data segment contained in the sub-data space and indicating whether said data segment has been mapped to the database storage disk;

(4) determining if a bit in said segment bit map is equal to a predetermined value, wherein said bit corresponds to the data segment of the sub-data space containing the page to be mapped, and said predetermined value indicates that the data segment corresponding to said bit has been mapped to the database storage disk;

(5) mapping a data segment of the data object from the database storage disk to the data segment of the sub-data space, wherein said data segment from the database storage disk contains the page to be mapped, and said mapping occurs if step (4) determines that said bit in said segment bit map does not equal said predetermined value; and

(6) setting said bit in said segment bit map to said predetermined value, wherein said setting occurs when step (5) is performed.

2. The method of claim 1, wherein the page to be mapped is to be modified, thereby creating a page to be modified, comprising the further steps of:

(7) determining whether said page to be modified requires a new mapping to the database storage disk;

wherein if step (7) determines that said page to be modified requires said new mapping to the database storage disk, the method comprise the further steps of:

(8) copying said page to be modified to another work space in computer memory and modifying said page to be modified in said another work space in computer memory;

(9) allocating a new database storage page in said database storage disk;

(10) determining whether a mapping request to map said page to be modified to said new database storage page in said database storage disk can be combined with a prior mapping request, wherein said prior mapping request maps a second page in the sub-data space to a second database storage page in said database storage disk and said page to be modified is adjacent to said second page in the sub-data space;

(11) issuing said prior mapping request if step (10) determines that said mapping request cannot be combined with said prior mapping request and said prior mapping request exists;

(12) copying said second page of said prior mapping request from said another work space in computer memory if step (11) issues said prior mapping request;

(13) preparing said mapping request if step (11) issues said prior mapping request or said prior mapping request does not exist; and

(14) combining said mapping request with said prior mapping request if step (10) determines that said mapping request can be combined with said prior mapping request.

3. A computer system for mapping on demand a page of a data object contained in a database and stored in one or more database storage disks, wherein the database comprises one or more database data objects, wherein the database is accessed via a contiguous data space representation, the contiguous data space being represented by a plurality of concatenated sub-data spaces, wherein each sub-data space is a virtual data space of a maximum size that is addressable by a computer operating system, each sub-data space comprising a plurality of data segments, each data segment comprising a plurality of pages, wherein the pages in the contiguous data space representation are contiguous, each page comprising a known number of addressable storage locations, wherein the contiguous data space, the sub-data spaces, the data segments, and the pages are addressable by a database management system, wherein a page to be mapped has been determined to be placed in a data segment of a sub-data space, comprising:

first determining means for determining whether the sub-data space has been created for representation in the contiguous data space;

sub-data space means for creating the sub-data space for representation in the contiguous data space if said first determining means determines that the sub-data space has not been created in the contiguous data space;

bit map means for creating a segment bit map for the sub-data space if said first determining means determines that the sub-data space has not been created in the contiguous data space, wherein said segment bit map comprises a plurality of bits, each bit representing a data segment contained in the sub-data space and indicating whether said data segment has been mapped to the database storage disk;

second determining means for determining if a bit in said segment bit map is equal to a predetermined value, wherein said bit corresponds to the data segment of the sub-data space containing the page to be mapped, and said predetermined value indicates that the data segment corresponding to said bit has been mapped to the database storage disk;

mapping means for mapping a data segment of the data object from the database storage disk to the data segment of the sub-data space, wherein said data segment from the database storage disk contains the page to be mapped, and said mapping occurs if said second determining means determines that said bit in said segment bit map does not equal said predetermined value; and

setting means for setting said bit in said segment bit map to said predetermined value, wherein said setting occurs when said mapping means is performed.

4. The computer system according to claim 3, wherein the page to be mapped is to be modified, thereby creating a page to be modified, further comprising:

third determining means for determining whether said page to be modified requires a new mapping to the database storage disk;

wherein if said third determining means determines that said page to be modified requires said new mapping to the database storage disk, the computer system further comprising:

first copying means for copying said page to be modified to another work space in computer memory and modifying said page to be modified in said another work space in computer memory;

allocating means for allocating a new database storage page in said database storage disk;

fourth determining means for determining whether a mapping request to map said page to be modified to said new database storage page in said database storage disk can be combined with a prior mapping request, wherein said prior mapping request maps a second page in the sub-data space to a second database storage page in said database storage disk and said page to be modified is adjacent to said second page in the sub-data space;

issuing means for issuing said prior mapping request if said fourth determining means determines that said mapping request cannot be combined with said prior mapping request and said prior mapping request exists;

second copying means for copying said second page of said prior mapping request from said another work space in computer memory if said issuing means issues said prior mapping request;

preparing means for preparing said mapping request if said issuing means issues said prior mapping request or said prior mapping request does not exist; and

combining means for combining said mapping request with said prior mapping request if said fourth determining means determines that said mapping request can be combined with said prior mapping request.

* * * * *

58/3,K/144 (Item 144 from file: 349)
PCT FULLTEXT

01054697
MICRO EDITION DYNAMIC OBJECT- DRIVEN DATABASE MANIPULATION AND MAPPING SYSTEM
SYSTEME DYNAMIQUE DE MICRO-EDITION PAR MANIPULATION ET MAPPAGE DE BASE DE
DONNEES A COMMANDE D'OBJETS
Patent Applicant/Assignee:
THOUGHT INC, 657 Mission Street, San Francisco, CA 94105, US, US (Residence), US (Nationality), (For all designated states except: US)
Patent Applicant/Inventor:
MULLINS Ward, 2222 Leavenworth Street, Apartment 304, San Francisco, CA
94133, US, US (Residence), US (Nationality), (Designated only for: US)
BOUCHER Robert, 68085 Valley Vista Drive, Cathedral City, CA 92234, US,
US (Residence), BR (Nationality), (Designated only for: US)
Legal Representative:
ORZECHOWSKI Karen Lee (agent), Liniak, Berenato & White, LLC, 6550 Rock Spring Drive, Suite 240, Bethesda, MD 20817, US,
Patent and Priority Information (Country, Number, Date):
Patent: WO 200383700 A1 20031009 (WO 0383700)
Application: WO 2003US8956 20030324 (PCT/WO US0308956)
Priority Application: US 2002367117 20020322
Designated States:
(Protection type is "patent" unless otherwise stated - for applications
prior to 2004)
AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU CZ DE DK DM DZ
EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR
LS LT LU LV MA MD MG MK MN MW MX MZ NI NO NZ OM PH PL PT RO RU SC SD SE
SG SK SL TJ TM TN TR TT TZ UA UG US UZ VC VN YU ZA ZM ZW
(EP) AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IT LU MC NL PT RO SE
SI SK TR

(OA) BF BJ CF CG CI CM GA GN GQ GW ML MR NE SN TD TG
(AP) GH GM KE LS MW MZ SD SL SZ TZ UG ZM ZW
(EA) AM AZ BY KG KZ MD RU TJ TM
Publication Language: English
Filing Language: English
Fulltext Word Count: 12869

International Patent Class (v7): G06F-017/30
Fulltext Availability:

US EQUIV - ATT'D

Detailed Description
Claims

Detailed Description
... above operates with a runtime library repository that can be modified
and tailored to optimize database access calls and to optimize the generation of SQL strings for a particular database. In a particularly preferred embodiment, the invention provides an object language software
program as described above, wherein the software program provides the ability to synchronize internal data and/or objects on the micro computer
or virtual machine emulator system with an external data store and to provide transparent persistence of data
and/or objects between the internal and external data stores,
In a particularly preferred embodiment, the invention provides an object
language software program as described above, wherein the synchronization
and transparent persistence is implemented through...hie. or are otherwise
available on the internet as publications,
1 hi one embodiment, the mapping information, rules, or metadata can be
maintained in a human-readable forniat and can be interpreted by an application to apply the rules to data objects. This information can be
stored directly in the data source or can be stored in...

...is called a "repository". This repository provides an easily maintainable and transportable format for the mapping 3o rules. Therulesandjavaobjectrelationshipscanbeupdated,orotherwisechanged, dynamically even while other rules or object relationships in the repository are being used to perform conversions. The rules and relationship definitions can be embedded in standard formats such as in
Java...

...proliferation of the rules to other sites where the rules can be used to
access objects (which can be bundled with the rules), used to derive other rules, etc. In a particularly preferred embodiment, this repository
is in an XML (extensible markup language) format, and the object model definitions can also be present in an ...CocoBase Enterprise Object/Relational Software package (hereafter CocoBase), which includes
the software tool CocoAdmin. Such updated software package, is available,
or is available shortly, from Thought, Inc., San Francisco, California.

An...

...CocoAdmin provides a mechanism for the export of maps defined against a
database into a modifiable XML format. This facility allows a system user
to select an existing map(s) to be exported, and to also specify the filename of the XML document to...

...edited using a standard text editor or XML editor, and can be modified
to reflect map custoinization requirements. The following discussion relates to specific features of the generated XML file.

The basic repository format has CBObj ect (CocoBase object class) definitions that reflect the select, insert, update, delete and call related map definitions for a CocoBase map. Each of those operations further consists of tables, fields and clauses
thatmayexisttospecifyhowtheobjectismappedtoandfromthedata source.

XML files contain a DTD (document type definition) entry which describes
the structure of the tags and data contained in the file. The DTD only checks for the existence of required elements, but...

...of those elements. Given this desired degree of freedom, it is possible
to enter invalid map information which will not be processed properly ...
to avoid improper editing. Ea general, modifications should be restricted
to the schema, table, and field name variables in the Tables and Fields
entries. These variables may require different values when they are exported from one database instance, and imported into another.

To begin exporting a set of maps, select File->Export YJ%AL Map Repository from the CocoAdmin pull-down menu. Multiple maps can be selected by holding down either the <Shift> or <Contro> when selecting the connection and map(s) to export.

When Next > is pressed in the dialog box provided, the list of selected maps will be presented in a list and the window will prompt for
a final acknowledgement of export.

The XML repository filenameCocoBase
CocoAdmin provides a mechanism for importing XML based CocoBase map definitions, or for importing an XMI object model definitions and then generating corresponding XML based CocoBase map definitions. Using the XML syntax defined in the thought
cocodemo3 t ier3 1

demos
resources
coco. dtd template file, CocoAdmin can import maps previously defined and exported from a different database or from a different object instance. The XML files generated from CocoAdmin will be validated against the DTD, and a...XML map definition from the CocoAdmin GUI select
File->Import XML Map Repository from the pull-down menu or Import XA11L Map Repository ftom the popup menu in the main CocoAdmin dialog. Import an XML based map definition by selecting the database connection
into which the import is to occur, and click the Browse button to find...
filename appears in the initial Import dialog. After the Next > button is pressed, the selected maps are compared with those already in the database. If a newer version of a map already exists in the database, by
default a Rag is set to retain the newer version. If the maps being imported are newer than the ones already in the database, then by default, a flag is set to import each of the newer ones. When...

...user can override any flags before the task is initiated. For example,
if the XML maps are older than the versions already in the database, this
condition causes the Already Exists? flag to be checked and the XML map
definitions will not be imported unless the user overrides this flag by
checking the Import/Overwrite? box.
When a user clicks the Import XML Repository button as described above,
maps marked for import will automatically be integrated into the list of
maps in the CocoBase...standard text editor can be used for this

operation. Most common edits consist of changing column names, table names,andlessfrequently,theschema namesofthetablesandthefields that the map may access.

The CocoBase Programmer's Guide available at the www.thoughtine.com website provides farther information about how to work with XML format maps and CocoBase, which published document is incorporated herein by reference. This document also explains how to set CocoBase (or CocoAdmin
of CocoBase) and its runtime modules to cache maps that an application will be using in order to speed up user access. Other user...

...For example, one URL option in CocoBase is to specify an XML repository

for the maps that an application will be using. Instead of using the database repository, the runtime can use an XML repository exclusively.
This teclu'lique allows CocoBase to execute against a production database
that cannot be modified with map definition tables. Optional mapping server plug-ins are also described.

Data source maps according to the present invention may be utilized to generate application programming code, ...most commercial 3o Application
Servers and Enterprise Java Bean Servers. A complete list of pre-configured targets can be viewed from the Connections window in the Generate Java wizard of CocoAdininwith data source specific nondelegation
database access code can be avoided by generating code, which delegates
the database access functions to CocoBase runtime libraries. Only CocoBase is believed to utilize this type of code that results in dynamic
O/R mapping system capabilities.

9 A computer system accordingmaps, or accessing objects on a system to make data changes related to a particular object and to promulgate the changes to that object as either local or global changes on the computer
system. I 10. A computer system according to claim 9, wherein the mapping
system portion provides an interface permitting an authorized user to edit or create the tables, fields, or attributes of a data map for an object as a table format or XML file format without requiring the user to
have extensive knowledge of a particular relational database as a source
of the data, or extensive knowledge about how to directly access that relational
database. I
I 11. A computer system according to any of claims I to 1 0, wherein the
mapping system provides an interface and features that permit a user to
access, create, or update the metadata of a map as a dynamic computer

system update, without requiring the user to either open a new connection
to the data source or to restart an object application program that is running while the user is dynamically evaluating or changing metadata for
a map, and wherein the metadata of a map that a user can dynamically evaluate or change includes a map description of data or relationships between data, and wherein such map description is at least one member selected from the group consisting of a data source relationship, a relationship between at least two objects of an object application, lo and both a data source relationship and a relationship between at least
two objects of an object application. i 12. A local or distributed computer system which is an intranet, internet or...a larger computer system. i 13. The computer system of claim 12, further comprising a mapping system wherein Object programming applications are tailored to delegate both the accessing of a data source and the generation of SQL strings to a runtime library repository, which repository can access the
database directly or through a database driver, such as a JDBC driver, without the need to embed specific database accessing mechanisms in the
application code, and wherein the persistence manager calls the mapping
system and in the call delegates accessing or updating of data in the data source...

...primary data source I I or record upon completion, including commit or

roll-back of changes for both data sources.

14 The computer system of claim 13, wherein ...and is not merely a duplicate of the micro computer system data source.

15 A object language software program that can run in a micro computer system according to any of...

...machine that emulates such a micro computer system, and can generate
application programming code from database maps and ...system, which delegates to a runtime library repository both of the functions of accessing a database and generating SQL strings that are specific to a database or to a JDBC driver for a type of database.
I
16 An object language software program according to claim 15, wherein the
runtime library repository can be modified and tailored to optimize database access calls and to optimize the generation of SQL strings for a
particular database. i 17. An object language software program according
to claim 16, wherein the software, program provides the ability to synchronize internal data and/or objects on the micro computer or virtual

machine emulator system with an external data store and to provide transparent persistence of data and/or objects between the internal and
external data stores.

18 An object language software program according to claim 17, wherein the
synchronization and transparent persistence is implemented...

US 20040006549A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2004/0006549 A1

Mullins et al. (43) Pub. Date: Jan. 8, 2004

(54) MICRO EDITION DYNAMIC OBJECT-DRIVEN DATABASE MANIPULATION AND MAPPING SYSTEM

(76) Inventors: Ward Mullins, San Francisco, CA (US); Robert Boucher, Cathedral City, CA (US)

Correspondence Address:
Robert G. Lev
4766 Michigan Boulevard
Youngstown, OH 44505 (US)

(21) Appl. No.: 10/396,216

(22) Filed: Mar. 24, 2003

Related U.S. Application Data

(60) Provisional application No. 60/367,117, filed on Mar. 22, 2002.

Publication Classification

(51) Int. Cl.[7] G06F 17/30

(52) U.S. Cl. 707/1

(57) ABSTRACT

The present invention provides a micro computer system and method for dynamic object-driven database manipulation and mapping system which relates in general to correlating or translating one type of database to another type of database or to an object programming application. Correlating or translating involves relational to object translation, object to object translation, relational to relational, or a combination of the above. Thus, the present invention is directed to dynamic mapping of databases to selected objects that can be used on both a micro computer system and in a larger computer system. Also provided are systems and methods that optionally include caching components, security features, data migration facilities, and components for reading, writing, interpreting and manipulating XML and XMI data files.

most appropriate data may not be accessed. Further, depending upon the amount of data available in a data store and the number of metadata maps available, the selection process may become very slow and cumbersome.

[0153] However, there is a definite need for translating between a wide variety of different data formats, other than just the relational to object translation of the basic system of U.S. Pat. No. 5,857,197. There is also a need for transmitting quickly and easily between various types of databases. Further, there is also a need for editing or otherwise altering data exchanged between databases for the purpose of facilitating object applications. The basic system of U.S. Pat. No. 5,857,197 needs further support to facilitate expeditious operation, both for its basic system and for modifications to its system that will accommodate better security and for selectivity of data.

[0154] Accordingly, there will be a substantial advantage in generating XML data documents for exchange between parties to carry out commercial transactions. Conversion to XML will also facilitate the use of data sources or data architecture other than relational databases. Examples of such mapping include object-to-object mapping and JAVA to object mapping. Once algorithms translating between XML and object formats are standardized, translations between any type of system can take place seamlessly and very quickly. Mapping can also include the use of JAVA Server Page (JSP) format.

[0155] Once editing has took place either for the metadata maps or even the production data from the datastore, changes can be recorded in the operation of snapshots so that a record of the map or other data at various times can be studied. As a result, data management paradigms can be developed. Normally such snapshots are made at the location of the party requesting the data (such as the object application user) once map modification is completed. The modified files can then be placed in a source code management system.

[0156] The dynamic mapping of the present invention can be applied where a Java object provides translation by mapping such an object to first data source (either relational or object) can also by mapping such an object to an XML for a second format database. This arrangement allows greater control as to how the data steps are exchanged, filtered, and validated by exchanging data.

[0157] While a number of preferred embodiments of the present invention have been discussed by way of example, the present invention is not to be limited thereby. Rather, the present invention is to be construed as including any and all variations, modifications, permutations, adaptations and embodiments that would occur to one skilled in this art once having been taught the present invention.

I claim:

1. A micro computer system comprising at least one data source and a mapping system wherein object programming applications are tailored to delegate both the accessing of a data source and the generation of SQL strings to a runtime library repository, which repository can access the database directly or through a database driver, such as a JDBC driver, without the need to embed specific database accessing mechanisms in the application code.

2. A system according to claim 1, wherein the micro computer system is an embedded device that is a portion of a larger system other than a traditional desktop or laptop computer.

3. A system according to claim 2, wherein the embedded system is a portable or cellular phone, an automobile computing system, part of a home electronic sound or entertainment system, a portable sound or entertainment system, an electronic security system, a smart video game console, or a combination thereof.

4. A system according to claim 1, wherein the micro computer system is a standalone computer system other than a traditional mainframe, desktop, or laptop system.

5. A system according to claim 4, wherein the standalone computer is smart mobile phone, a personal organizer, a portable stock market trading device, a hand held computer or a PDA.

6. A system according to claim 5, wherein standalone computer is adaptable to or includes portable internet access.

7. A system according to claim 1, having a concurrent parameter setting mechanism wherein the runtime library repository can be set to access a particular data source and to generate data source specific database calls and SQL strings.

8. A computer system according to claim 1, wherein the mapping system portion is designed to provide different maps for particular objects to different users and does not permit direct access of computer system users to JDBC drivers for any mapped data sources of the computer system, and wherein the mapping system provides varying levels of access to the mapped data sources for at least two different users of the same system, whereby a user only has access to a particular list of maps that are available to the security level of that user.

9. A computer system according to claim 1, wherein the mapping system portion is designed to provide to a system user who is accessing, creating or updating maps, or accessing objects on a system to make data changes related to a particular object and to promulgate the changes to that object as either local or global changes on the computer system.

10. A computer system according to claim 9, wherein the mapping system portion provides an interface permitting an authorized user to edit or create the tables, fields, or attributes of a data map for an object as a table format or XML file format without requiring the user to have extensive knowledge of a particular relational database as a source of the data, or extensive knowledge about how to directly access that relational database.

11. A computer system according to claim 1, wherein the mapping system provides an interface and features that permit a user to access, create, or update the metadata of a map as a dynamic computer system update, without requiring the user to either open a new connection to the data source or to restart an object application program that is running while the user is dynamically evaluating or changing metadata for a map, and wherein the metadata of a map that a user can dynamically evaluate or change includes a map description of data or relationships between data, and wherein such map description is at least one member selected from the group consisting of a data source relationship, a relationship between at least two objects of an object

application, and both a data source relationship and a relationship between at least two objects of an object application.

12. A local or distributed computer system which is an intranet, internet or satellite communicated system, wherein the local or distributed system includes a micro computer system or an embedded system comprising a fully synchronized data system and an O/R repository that permits persistence and synchronizing of data between a data source on the micro computer system and a main data source on a larger computer system.

13. The computer system of claim 12, further comprising a mapping system wherein object programming applications are tailored to delegate both the accessing of a data source and the generation of SQL strings to a runtime library repository, which repository can access the database directly or through a database driver, such as a JDBC driver, without the need to embed specific database accessing mechanisms in the application code, and wherein the persistence manager calls the mapping system and in the call delegates accessing or updating of data in the data source to the repository layer, and wherein the a persistence monitor does not complete a transaction until it is notified by the mapping system that a data source has been updated by the mapping system such that any changes to the data will automatically be synchronized with the primary data source or record upon completion, including commit or roll-back of changes for both data sources.

14. The computer system of claim 13, wherein at least one temporary data source can be set up on a microcomputer system or embedded system and is synchronized with the persistence monitor with a data source on a larger system wherein the larger system can be accessed by a JDBC driver and is not merely a duplicate of the micro computer system data source.

15. A object language software program that can run in a micro computer system according to claim 1, or in a virtual machine that emulates such a micro computer system, and can generate application programming code from database maps and thereby provide a programming application that will run on a micro computer system, which delegates to a runtime library repository both of the functions of accessing a database and generating SQL strings that are specific to a database or to a JDBC driver for a type of database.

16. An object language software program according to claim 15, wherein the runtime library repository can be modified and tailored to optimize database access calls and to optimize the generation of SQL strings for a particular database.

17. An object language software program according to claim 16, wherein the software program provides the ability to synchronize internal data and/or objects on the micro computer or virtual machine emulator system with an external data store and to provide transparent persistence of data and/or objects between the internal and external data stores.

18. An object language software program according to claim 17, wherein the synchronization and transparent persistence is implemented through communication between directly connected systems or by indirectly communicated machines via telephone, intranet, internet, or satellite communication, or through a combination thereof.

* * * * *

58/3,K/210 (Item 210 from file: 349)
PCT FULLTEXT

00501653 **Image available**
OBJECT-RELATIONAL MAPPING TOOL THAT PROCESSES VIEWS
OUTIL DE MAPPAGE OBJET/RELATION POUR LE TRAITEMENT DE VUES
Patent Applicant/Assignee:
SHARMA Rahul,
DEMICHIEL Linda G,
Inventor(s):
SHARMA Rahul,
DEMICHIEL Linda G,
Patent and Priority Information (Country, Number, Date):
Patent: WO 9933005 A1 19990701
Application: WO 98US27246 19981221 (PCT/WO US9827246)
Priority Application: US 9768415 19971222; US 98106186 19980629
Designated States:
(Protection type is "patent" unless otherwise stated - for applications
prior to 2004)
AL AM AT AU AZ BA BB BG BR BY CA CH CN CU CZ DE DK EE ES FI GB GD GE GH
GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MD MG MK MN
MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT UA UG UZ VN YU ZW
GH GM KE LS MW SD SZ UG ZW AM AZ BY KG KZ MD RU TJ TM AT BE CH CY DE DK
ES FI FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM GA GN GW ML MR NE
SN TD TG
Publication Language: English
Fulltext Word Count: 7556

Fulltext Availability:
Detailed Description
Claims

Detailed Description
... being opened, the object-relational mapping tool proceeds to state 3
12.

However, if the object-relational mapping tool determines that the request received is to import a relational database schema, the object -relational mapping tool imports the schema (state 304). In this state,
the mapping tool imports the database schema using the Java database connectivity API (JDBQ available from Sun Microsystems of Palo Alto, California. JDBC is a Java...

...and interfaces written in the Java programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API. JDBC is described in greater detail in Hamilton, Cattell, and Fisher, JDBC Database Access with Java, AddisonWesley (I 997), which is incorporated herein by reference.

When a database schema is imported, database data structures are created to reflect both the views and the tables found in the database schema. For example, if the database schema II 8 depicted in Figure 2 was imported, the object-relational mapping tool would create seven data structures, five data structures for each of the five views 206, 208, 210, 212, and 214 and two data...

...dependency.

After a database schema has been imported, the mapping tool deten-nines if automatic mapping has been previously requested (state 306). The mapping tool contains an edit menu which allows the programmer to specify if automatic mapping is to take place during the importation process. There are three types of automatic mapping performed by the mapping tool.

In the first case, the mapping tool will automatically map all tables and all views. In the second case, the mapping tool will automatically map only the base tables and not any views. In the third case, the mapping tool will automatically map only top level views and tables, in which case only tables and views upon which there are no dependencies will be mapped. With this latter option, the mapping tool maps the highest level view of a table or the table itself if there are no views defined on that table.

For example, if the database schema II 8 depicted in Figure 2 were imported, and the automatic mapping feature were previously set to map all views and tables (i.e., type one), the object-relational mapping tool would automatically map the data structures associated with all five views, 206, 208, 210, 212, and 214 and both tables, 202 and 204. On the other hand, if automatic mapping feature was previously set to map only top level views and tables (i.e., type three), the object-relational

mapping tool would only automatically map the data structures associated
with West Coast Employee view 214 and the Employee Withholding view 212.

If the automatic mapping feature is set in state 306, the object -relational mapping tool generates an object model reflecting each data
structure that is to be automatically mapped (state 308). The object model 1 16 represents an intermediate form of the information for a class
before it is...

...information. Thus, if a particular table or view is to be mapped, the
object-relational mapping tool will create a corresponding object in the
object model. If, on the other hand, a particular table or view is to be
unmapped, then the object-relational mapping tool will delete the corresponding object in the object model.

In addition, based on the attributes of a particular database structure,
the object-relational mapping tool will set various flags that serve to
notify the object-relational mapping tool to generate methods corresponding to the flags when the source code is generated. Thus...

...mapping tool will set a flag so that modifier methods for the view's
corresponding field(s) in the mapped class will be generated in the source code.

Once the object model has been created, the object-relational mapping tool creates a source code file containing classes represented by the object in the object model (state 3 1 0). In particular, the object -relational mapping tool will generate classes with methods and data members that represent the object model. Thus, for instance, if a particular mapped object, representing a view, is read-only, then only "get methods" for all fields of...

...to read-only, the object-relational mapping tool will only create "get
methods" for each field of the class. Also, the modifier and accessor methods for these classes can be enabled or disabled selectively as part
of the mapping custornization process.

The states perforined by the object-relational mapping tool when generating a source code file from an object model, the importation of a

database schema, and the generation of an object model from an imported
database schema, are described in greater detail in co-pending U.S. ...An Integrated Graphical User Interface Method and Apparatus for Object-to-Database and Database-to-Object Mapping" which has previously
been incorporated by reference and in co-pending U.S. Application No. ,
entitled "Integrating Both Modifications To An Object Model And Modifications To A Database Into Source Code By An Object-Relational Mapping Tool" which has previously been incorporated by reference.

After the source code is automatically generated...

...or if the mapping tool is merely opening a previously saved imported
database schema, the object-relational mapping tool displays the mapping
status in a user-interface mapping

US006175837B1

(12) **United States Patent**
Sharma et al.

(10) **Patent No.: US 6,175,837 B1**
(45) **Date of Patent: Jan. 16, 2001**

(54) **OBJECT-RELATIONAL MAPPING TOLL THAT PROCESSES VIEWS**

(75) Inventors: **Rahul Sharma**, Santa Clara; **Linda G. DeMichiel**, Los Altos, both of CA (US)

(73) Assignees: **Sun Microsystems, Inc.**, Palo Alto, CA (US); **Baan Development, B.V.**, Barneveld (NL)

(*) Notice: Under 35 U.S.C. 154(b), the term of this patent shall be extended for 0 days.

(21) Appl. No.: **09/106,186**

(22) Filed: **Jun. 29, 1998**

Related U.S. Application Data

(60) Provisional application No. 60/068,415, filed on Dec. 22, 1997.

(51) **Int. Cl.**[7] **G06F 17/30**
(52) **U.S. Cl.** **707/103**; 707/102
(58) **Field of Search** 707/103, 102

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,589,092	5/1986	Matick	711/207
5,043,876	8/1991	Terry	707/201
5,261,069	11/1993	Wilkinson et al.	711/145
5,263,155	11/1993	Wang	707/8
5,280,612	1/1994	Lorie et al.	707/8
5,291,583	3/1994	Bapat	717/5
5,301,297	4/1994	Menon et al.	711/114
5,499,371 *	3/1996	Henninger et al.	717/2
5,504,885	4/1996	Alashqur	717/5
5,542,078	7/1996	Martel et al.	707/101
5,574,882	11/1996	Menon et al.	711/114
5,596,746	1/1997	Shen et al.	707/101
5,659,738 *	8/1997	Letkeman et al.	707/102
5,701,453 *	12/1997	Maloney et al.	707/2
5,732,257	3/1998	Atkinson et al.	707/4
5,742,813	4/1998	Kavanagh et al.	707/8
5,765,159	6/1998	Srinivasan	707/102
5,774,731	6/1998	Higuchi et al.	710/200
5,829,006	10/1998	Parvathaneny et al.	707/101
5,835,910	11/1998	Kavanagh et al.	707/103
5,850,544	12/1998	Parvathaneny et al.	707/101
5,857,197	1/1999	Mullins	707/103
5,878,411 *	5/1999	Burroughs et al.	707/4
5,878,419	3/1999	Carter	707/10
5,893,108	4/1999	Srinivasan et al.	707/103
5,907,846	5/1999	Berner et al.	707/103
5,937,402 *	8/1999	Pandit	707/4
5,937,409	8/1999	Wetherbee	707/103
6,003,040	12/1999	Mital et al.	707/103
6,009,428	12/1999	Kleewein et al.	707/10
6,038,565	3/2000	Nock	707/101
6,049,673	4/2000	McComb et al.	717/11

FOREIGN PATENT DOCUMENTS

0 472 812 A1	2/1991	(EP) .
WO 95/03586	2/1995	(WO) .
WO 95/04960	2/1995	(WO) .
WO 97/03406	1/1997	(WO) .

OTHER PUBLICATIONS

Fitsilis et al., "Producing Database Schemata from an Object Oriented Design", IEEE, pp. 251–257 (1994).

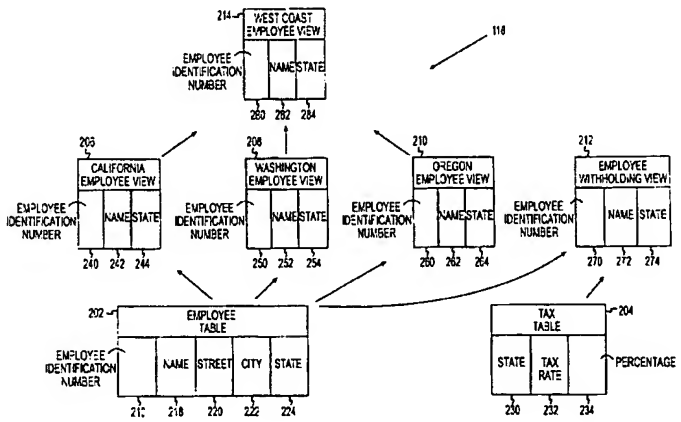
(List continued on next page.)

Primary Examiner—Thomas G. Black
Assistant Examiner—William Trinh
(74) *Attorney, Agent, or Firm*—Finnegan, Henderson, Farabow, Garrett & Dunner, L.L.P.

(57) **ABSTRACT**

An object-relational mapping tool that can process views is provided. In processing views, the object-relational mapping tool identifies the dependency relationship between the various views and tables and allows the programmer to customize the mapping that will occur by selectively determining which tables and views get mapped. This customization allows the mapping tool to map an individual table or view or to map (1) all tables and views, (2) all tables only, and (3) tables and views for which there are no dependencies. Also, the mapping tool allows the programmer to view and set the updatability of a particular view or table.

26 Claims, 9 Drawing Sheets



methods) can be selectively overridden by the programmer by interfacing with the class information window **502** in FIG. **5**, which was discussed above.

If the programmer did not choose to modify the updatability of a particular table or view, or if the programmer was unable to modify the updatability of a particular table or view, or if the programmer successfully modified the updatability of a particular table or view, the programmer can now determine whether or not to manually map a particular table or view (state **326**). If the programmer decides not to manually map a particular view or table, processing ends.

However, once the programmer decides to manually map a table or a view in state **326**, the programmer must next select the table or view to map (state **328**). A table or view is manually mapped by interacting with the user-interface mapping display window **400** depicted in FIG. **4**. First, the programmer selects a particular table or view (e.g. view **418**). By selecting a particular table or view, the user-interface mapping tool highlights the entire row of information associated with the selected table or view (e.g. view **418**). Next, the programmer simply selects the selection button (**417**) associated with the highlighted row (e.g. view **418**). Once the appropriate selection is made and the object-relational mapping tool produces a selection box **424**, the programmer simply toggles the selection **424**, thereby instructing the object-oriented mapping tool to either map or unmap the selected table or view.

In response to the manual mapping request in state **326**, the object-relational mapping tool determines whether the request is to map a table or to map a view (state **328**). If the programmer requests to map a view, the object-relational mapping tool displays a view dependency window (state **330**). The view dependency window **702** depicted in FIG. **7** is automatically displayed by the object-relational mapping tool when the selected view **704** that is about to be mapped has dependent views **706** or dependent base tables **708** that are already mapped. The view dependency window allows the programmer, in conjunction with mapping the selected view **712** to unmap the dependent views and tables or keep their status unchanged **710**. Once the programmer determines whether or not to unmap or leave unchanged the dependent views and tables, the programmer selects the OK button **712** to initiate the mapping procedure.

On the other hand, if in state **328** the programmer was attempting to manually map a table, the object-relational mapping tool, in response to the request, will display a table dependency window (state **332**). The table dependency window **802** depicted in FIG. **8** will automatically be displayed by the object-relational mapping tool when the selected table **804** about to be mapped has dependent views that are already mapped **806**. The table dependency display window allows the programmer to unmap all the views or keep their status unchanged **808**. Once the programmer determines whether or not to unmap or leave unchanged the dependent views, the programmer selects the OK button **810** to initiate the mapping procedure.

Once the programmer manually selects a view or table to be mapped, the object-relational mapping tool generates, as explained above in state **308**, an object model reflecting the table or view's data structure to be mapped (state **334**). After the object model has been created, the object-relational mapping tool creates, as explained above in state **310**, a source code file containing the class represented by the object in the object model (state **336**).

After the object-relational mapping tool maps the selected view or table, the programmer determines if any more views or tables require mapping (state **338**). If no more tables or views require mapping, processing ends, otherwise the programmer selects a new table or view to be manually mapped (state **328**).

Conclusion

The foregoing description of an implementation of the invention has been presented for purposes of illustration and description. It is not exhaustive and does not limit the invention to the precise form disclosed. Modifications and variations are possible in light of the above teachings or may be acquired from practicing the invention. For example, the described implementation includes software but the present invention may be implemented as a combination of hardware and software or in hardware alone. The scope of the invention is defined only by the claims and their equivalents.

What is claimed is:

1. A method in a data processing system containing a database with a logical structure comprising tables and views, wherein the tables and views form a hierarchy, the method comprising:

importing the logical structure of the database;

displaying the hierarchy to a user;

receiving an indication of a view, based on user input; and

generating source code that reflects at least one portion of the logical structure of the database including at least the indicated view.

2. The method of claim **1** wherein the generating includes:

generating source code reflecting the tables and the views.

3. The method of claim **1** wherein the generating includes:

generating source code reflecting the views.

4. The method of claim **1** wherein the generating includes:

receiving user input indicating an additional portion of the logical structure; and

generating source code that reflects the at least one portion and the additional portion.

5. The method of claim **1** wherein the generating includes:

receiving user input indicating that the at least one portion should not be reflected in the source code; and

generating source code that does not reflect the at least one portion.

6. A method in a data processing system containing a database with a logical structure comprising tables and views, wherein a plurality of the tables and views do not have dependencies, the method comprising:

importing the logical structure of the database;

automatically generating source code reflecting only the plurality of the tables and the views that do not have dependencies.

7. A data processing system containing a database with a logical structure comprising tables and views, comprising:

means for importing the logical structure of the database;

means for displaying a hierarchy of the logical structure to a user;

means for selecting a table or view, responsive to user input;

means for displaying a definition of a view, responsive to user input; and

means for editing an updatability status of a view, responsive to user input.

8. A method in a data processing system with a database having a logical structure with views and tables, comprising:

importing the logical structure of the database;

creating a data structure representing the logical structure of the database;

creating from the data structure an object model containing information reflecting at least one portion of the

logical structure including information reflecting at least one view;

creating from the object model source code reflecting the at least one portion of the logical structure;

displaying the imported logical structure to a user;

receiving modifications to the at least one portion from the user;

updating the object model to reflect the modifications; and

generating from the updated object model new source code to reflect the modifications.

9. The method of claim **8**, wherein the at least one portion has a customizable updatability, and wherein the receiving modifications includes:

receiving a request to change the updatability status of the at least one portion of the logical structure, and wherein the updating the object model includes:

checking the relational model for consistency of changed updatability; and

updating the object model to reflect the change in the updatability status.

10. The method of claim **8** wherein the receiving modifications includes:

receiving an indication of an additional portion of the logical structure, wherein the updating the object model includes:

creating a portion of the object model to reflect the additional portion, and wherein the generating includes:

generating the new source code to reflect the additional portion.

11. The method of claim **8** wherein the receiving modifications includes:

receiving an indication to not generate source code for the at least one portion, wherein the updating the object model includes:

creating the object model such that it does not reflect the at least one portion, and wherein the generating includes:

generating the new source code to such that it does not reflect the at least one portion.

12. A data processing system comprising:

a secondary storage device containing a database with a logical structure of tables and views, wherein the tables and views have an associated updatability status;

a memory containing an object-relational mapping tool configured to import at least one portion of the logical structure including at least one view and configured to generate source code reflecting the at least one portion of the logical structure; and

a processor configured to run the object-relational mapping tool, wherein the object-relational mapping tool further includes:

a display component for displaying the updatability status to a user; and

an updatability modification component configured to modify the updatability status of the at least one portion of the logical structure, responsive to user input.

13. The data processing system of claim **12** wherein the object-relational mapping tool contains a manual mapping component configured to operate after the source code has been generated and configured to update the source code to reflect an additional portion of the logical structure responsive to user input.

14. The data processing system of claim **12** wherein the object-relational mapping tool contains a manual mapping component configured to operate after the source code has been generated and configured to update the source code so that it does not reflect the at least one portion of the logical structure responsive to user input.

15. A data processing system with a database having a logical structure with views and tables, comprising:

means for importing the logical structure of the database;

means for creating a data structure representing the logical structure of the database;

means for creating from the data structure an object model containing information reflecting at least one portion of the logical structure including information reflecting at least one view;

means for creating from the object model source code reflecting the at least one portion of the logical structure;

means for displaying the imported logical structure to a user;

means for receiving modifications to the at least one portion from the user;

means for updating the object model to reflect the modifications; and

means for generating from the updated object model new source code to reflect the modifications.

16. A computer-readable medium containing instructions for controlling a data processing system to perform a method, the data processing system containing a database with a logical structure comprising tables and views, wherein a plurality of the tables and views do not have dependencies, the method comprising:

importing the logical structure of the database;

automatically generating source code only reflecting the plurality of the tables and the views that do not have dependencies.

17. A method in a data processing system containing a database with a logical structure comprising tables and views, comprising:

importing the logical structure of the database;

displaying a hierarchy of the logical structure to a user;

selecting a table or view, responsive to user input;

displaying a definition of a view, responsive to user input; and

editing an updatability status of a view, responsive to user input.

18. A computer-readable medium containing instructions for controlling a data processing system to perform a method, the data processing system containing a database with a logical structure comprising tables and views, wherein the tables and views form a hierarchy, the method comprising:

importing the logical structure of the database;

displaying the hierarchy to a user;

receiving an indication of a view, based on user input; and

generating source code that reflects at least one portion of the logical structure of the database including at least the indicated view.

19. The computer-readable medium of claim **18** wherein the generating includes:

generating source code reflecting the tables and the views.

20. The computer-readable medium of claim **18** wherein the generating includes:

generating source code reflecting the views.

21. The computer-readable medium of claim **18** wherein the generating includes:

receiving user input indicating an additional portion of the logical structure; and

generating source code that reflects the at least one portion and the additional portion.

22. The computer-readable medium of claim **18** wherein the generating includes:

receiving user input indicating that the at least one portion should not be reflected in the source code; and

generating source code that does not reflect the at least one portion.

23. A method in a data processing system containing a database with a logical structure comprising tables and views, the method comprising:

importing the logical structure of the database;

displaying the logical structure to a user;

selecting a table or view, responsive to user input;

displaying mapping fields for each table and view to a user, wherein the mapping fields indicate a mapping status of each corresponding table and view of the logical structure;

editing the mapping field for the selected table or view, responsive to user input; and

displaying the edited mapping field to the user.

24. The method of claim **23**, wherein the step of displaying mapping fields further includes:

displaying the hierarchy of the selected table or view.

25. A data processing system containing a database with a logical structure comprising tables and views, comprising:

means for importing the logical structure of the database;

means for displaying the logical structure to a user;

means for selecting a table or view, responsive to user input;

means for displaying mapping fields for each table and view to a user; wherein the mapping fields indicate a mapping status of each corresponding table and view of the logical structure;

means for editing the mapping field for the selected table or view, responsive to user input; and

means for displaying the edited mapping field to the user.

26. The data processing system of claim **25**, wherein the means for displaying mapping fields further includes means for displaying the hierarchy of the selected table or view.

* * * * *